CAMPUS
DE EXCELENCIA
INTERNACIONAL

Grado en Ingeniería Informática

Universidad Politécnica de Madrid

Escuela Técnica Superior
de Ingenieros Informáticos

TRABAJO DE FIN DE GRADO

Uso de Leap Motion en juegos didácticos para niños con necesidades educativas especiales

Autor: Iván Jesús García Agenjo

Director: Ángel Lucas González Martínez

MADRID, JUNIO DE 2015

ÍNDICE

ÍNDICE i
RESUMEN iii
ABSTRACT iv
1. INTRODUCCIÓN Y OBJETIVOS 1
1.1. Objetivos 2
2. Trabajos Previos 3
2.1. Interacttico 3
2.2. Leap Motion 6
3. ESTADO DEL ARTE 9
3.1. Lenguajes de programación 9
3.1.1. C#: 9
3.1.2. ActionScript: 9
3.2. SDK 10
3.3. Reconocedor de Leap Motion 13
3.3.1. Esqueleto 13
3.3.2. Definición de gestos 14
3.3.3. Reconocimiento de gestos 18
3.4. Definición de la información necesaria para el estudio de la interacción con Leap Motion 23
4. DESAROLLO 25
4.1. Requisitos del sistema 25
4.1.1. Requisitos del reconocedor Leap Motion 25
4.1.2. Requisitos de los gestos 26
4.2. División del proyecto 28
4.2.1. C# 28
4.2.2. ActionScript 29
4.2.3. Wrapper 29
4.3. Actualización del Reconocedor 30
4.3.1. Esqueleto 30
4.3.2. Grabador de gestos 32
4.3.3. Reconocimiento de gestos 32
4.4. Grabación de los Gestos 33
4.5. Inserción del reconocedor en el juego Interacttico 35

4.5.1. C# 35
4.5.2. Flash 36
4.5.3. ActionScript 37
4.6. Generación del ejecutable 38
4.7. Pruebas 40
4.7.1. Pruebas unitarias 40
4.7.2. Pruebas de sistema 41
5. CONCLUSIONES 42
6. FUTURAS LÍNEAS DE TRABAJO 44
6.1. Recogida de información sobre Leap Motion 44
6.2. Actualización de Leap Motion 44
6.3. Nuevos dispositivos de interacción 46
6.4. Nuevos mini juegos 46
7. BIBLIOGAFRÍA 47
8. Anexo: Estructura de directorios 48
8.1. Estructura del prototipo básico de Leap Motion 48
8.2. Estructura Interacttico 49

RESUMEN

Este Trabajo de Fin de Grado (TFG) tiene el objetivo incorporar el dispositivo Leap Motion [1] en un juego educativo para niños con necesidades educativas especiales para permitirles aprender de una forma divertida mientras disfrutan con los mini juegos que ofrece nuestra aplicación.

Está destinado al apoyo del sistema educativo para los niños con necesidades educativas especiales. Debido al público que tenemos como objetivo debemos de tener en cuenta que hay distintos tipos de usuarios según el tipo de discapacidad que tienen. Entre ellas tenemos discapacidad visual, auditiva, cognitiva y motriz.

Tenemos distintos mini juegos para facilitar el aprendizaje de las letras y nuevas palabras, los nombres de colores y diferenciarlos y la asociación de conceptos mediante ejemplos sencillos como son ropa, juguetes y comida. Para hacer que la interacción sea más divertida tenemos distintos tipos de dispositivos de interacción: unos comunes como son el teclado y la pantalla táctil y otros más novedosos como son Kinect [2] y Leap Motion que es el que se introducirá en el desarrollo de este Trabajo de Fin de Grado.

El otro objetivo de este proyecto es el estudio de los distintos dispositivos de interacción. Se quiere descubrir qué tipo de sistemas de interacción son más sencillos de aprender, cuáles son más intuitivos para los niños, los que les resultan más interesantes permitiendo captar mejor su atención y sus opuestos, es decir, los que son más difíciles de entender, los más monótonos y los más aburridos para ellos.

ABSTRACT

This Final Degree Project (TFG) aims to incorporate the Leap Motion device [1] in an educational game for children with special educational needs to enable them to learn in a funny way while enjoying the mini games that our application offered.

It is intended to support the education system for children with special educational needs. Because the public that we have as objective we must take into account that there are different types of users depending on the type of disability they have. Among them we have visual, auditory, cognitive and motor disabilities.

We have different mini games to make easier learning of letters and new words, names and distinguish colors and the association of concepts through simple examples such as clothing, toys and food. To make the interaction more fun we have different interaction devices: common such as the keyboard and the touch screen and other more innovative such as Kinect [2] and Leap Motion which is to be introduced in the development of this Final Degree Work.

The other objective of this project is to study the various interaction devices. You want to find out what type of interaction systems are easier to learn, which are more intuitive for children, who are more interesting allowing better capture their attention and their opposites, that is, those that are more difficult to understand, the most monotonous and most boring for them.

1. INTRODUCCIÓN Y OBJETIVOS

Actualmente hay varios dispositivos de interacción que permiten a las personas interactuar con los ordenadores. Esta interacción se ha ido desarrollando con el paso del tiempo para mejorar la experiencia de los usuarios intentando hacer que la forma de interactuar con el ordenador sea lo más natural e intuitiva posible.

El objetivo de este proyecto es la inclusión del dispositivo Leap Motion [1] en un juego educativo para niños con necesidades educativas especiales. En el proyecto del juego disponemos de distintos mini juegos (Figura 1) para facilitar el aprendizaje de las letras, palabras, colores y asociación de conceptos. Dentro de estos minijuegos ya se permite el uso de distintos dispositivos de interacción como son el teclado, Kinect [2] y la pantalla táctil. También se decidió introducir este dispositivo que permite interactuar con el ordenador mediante la realización de gestos en el aire con las manos y los dedos. Esto nos permitirá evaluar la idoneidad de este dispositivo. Se espera hacer pruebas en el futuro con niños en sus escuelas con el apoyo de la Consejería de Educación de la Comunidad Autónoma de Madrid.

Figura 1 Minijuegos

Leap Motion es el dispositivo de interacción que queremos introducir (Figura 2). Este dispositivo funciona mediante tres cámaras infrarrojas localizadas en un pequeño hardware que permiten detectar las manos del usuario y obtener su posición relativa. De esta forma el usuario es capaz de interactuar con el ordenador simplemente moviendo las manos encima del dispositivo. Este dispositivo dispone de su propia tienda de aplicaciones que ofrecen

soluciones para múltiples campos: puede emular una pantalla táctil o permite movimientos similares a los de un *touchpad* entre otros.

Figura 2 Dispositivo Leap Motion

1.1. Objetivos

El principal objetivo dentro de todo el ámbito en el que se desarrolla este sistema es facilitar el aprendizaje para los niños con necesidades educativas especiales, estando a su vez entretenidos, motivados y divirtiéndose. Por eso se ha creado esta aplicación que funciona mediante mini juegos y empleando dispositivos de interacción que les permitan interactuar de una forma nueva para que sea ameno y divertido.

Otro objetivo es el estudio sobre los sistemas de interacción con estos dispositivos, para este Trabajo de Fin de Grado en particular hay que centrarse en el dispositivo Leap Motion, que funciona mediante el reconocimiento de las manos en el aire. Al ser un dispositivo poco común se espera que resulte llamativo para los niños que podrán jugar moviendo sus manos, una forma de interacción poco común para comunicarse con un ordenador.

También está propuesto como objetivo el entrenamiento de los niños en el uso de los gestos que se van a usar con los dispositivos como son la pantalla táctil, el teclado, Kinect y Leap Motion.

2. Trabajos Previos

2.1. Interacttico

Interacttico es el nombre del proyecto del laboratorio de investigación CETTICO que generó el juego didáctico para niños con necesidades educativas especiales. Durante el primer semestre del curso 2014-2015 estuve realizando el Practicum en dicho laboratorio en el proyecto Interacttico [3]. Mi trabajo consistía en la recogida de información de los distintos dispositivos de interacción que empleamos en el juego y almacenarla en la base de datos para facilitar su posterior estudio. El proyecto tiene dos objetivos fundamentales que son facilitar el aprendizaje a los niños con necesidades educativas especiales y el estudio de los dispositivos de interacción.

En el proyecto se emplean múltiples juegos educativos con los que los niños aprenden mientras interactúan con nuevos dispositivos. Entre los juegos disponibles tenemos:

- Bolos (Figura 3): Diseñado para el aprendizaje de los colores. Se debe seleccionar la bola de la nube que tenga el mismo color de la primera línea de bolos para poder derribarlos.

Figura 3 Bolos

- Deletrea (Figura 4): Pensado para facilitar el estudio de las letras y las palabras. Al abrir el cobre aparecerá una palabra que deberán escribir seleccionando las letras en el orden correcto desde la nube.

Figura 4 Deletrea

- Asociación según el nivel de dificultad (Figura 5): Mini juego que facilita relacionar conceptos similares. En este caso tiene que seleccionar de la nube un elemento que pertenezca al mismo concepto (ropa o juguete en este caso) que el de la parte abierta de la caja e introducirlo en ella.

Figura 5 Fácil

Uno de los elementos de interés de este proyecto es permitir diferentes dispositivos de interacción durante el desarrollo del juego. Al iniciar el mini juego se pregunta qué dispositivo se quiere emplear (Figura 6). Al entrar en el proyecto entre los dispositivos disponibles se encontraban el Kinect, la pantalla táctil y el teclado.

Figura 6 Dispositivos Disponibles

Durante el desarrollo de mi proyecto también se está trabajando para introducir el nuevo dispositivo de Microsoft que salió con la consola XBOX ONE llamado Kinect V2, que emplea un nuevo SDK que proporciona una API que no es compatible con el Kinect anterior. Además se quiere introducir el dispositivo Leap Motion que permite el reconocimiento de las manos en el aire (Figura 7) para interactuar con los mini juegos. Esta última parte es el objetivo principal de este trabajo de fin de grado.

Figura 7 Reconocimiento de manos [4]

2.2. Leap Motion

Para el desarrollo de este proyecto he de mencionar a Karim Laazizi Ruiz [5], que es mi predecesor en este proyecto. Su proyecto de fin de carrera presentado en junio de 2014 basado en el desarrollo de un prototipo básico que estudiara el uso del dispositivo Leap Motion y comprobar que se podían definir y reconocer gestos ha supuesto la base principal para mi proyecto. Logró realizar un prototipo funcional empleando el SDK 1.6 que había disponible para Leap Motion en el momento que comenzó su proyecto. A partir de su trabajo mi tarea era actualizar su prototipo con el último SDK disponible para mejorar el reconocimiento e incorporarlo al proyecto de interacción Interacttico.

El objetivo su proyecto era permitir a una persona que tiene una aplicación para PC y quiere interactuar mediante el Leap Motion permitirle hacerlo con facilidad. Simplemente tiene que grabar los gestos (Figura 8) que quiere emplear para controlar su aplicación con el grabador de gestos e integrar las respuestas obtenidas por el reconocedor. Para ello tendrá que grabar un gesto representativo para cada acción que requiera su aplicación. Si por ejemplo tiene un menú tendrá que utilizar tres gestos: elemento siguiente del menú, elemento anterior del menú y seleccionar elemento. Al realizar un gesto se realiza la acción asociada a la tecla que sustituye, haciendo que integrar Leap Motion sea sencillo.

Figura 8 Grabador Leap Motion

El proyecto está pensado para ser lo más personalizable posible. Para ello al crear un gesto el usuario tiene una serie de parámetros que puede definir a su gusto o dejarlos por defecto. Entre estos parámetros se encuentran la velocidad de movimiento, en qué ejes se realiza el

movimiento (dirección) o el rango de similitud que tiene que tener el gesto reconocido con el guardado. De esta forma se le permite al usuario que un mismo gesto con distintas velocidades haga distintas acciones. Mediante los ejes permites hacer un movimiento sin tener en cuenta la altura o la profundidad de la mano como podría ser el movimiento "derecha".

La aplicación del usuario tendrá almacenados los datos de los gestos que necesita tras haberlos grabado. El reconocedor comparará los gestos que se realizan mientas la aplicación está abierta con los gestos almacenados. Si se reconoce el gesto se lanza a la aplicación un evento con el gesto recibido, la aplicación deberá actuar de la forma que le corresponda tras recibir el gesto. En la Figura 9 se muestra un esquema para facilitar la comprensión del reconocedor.

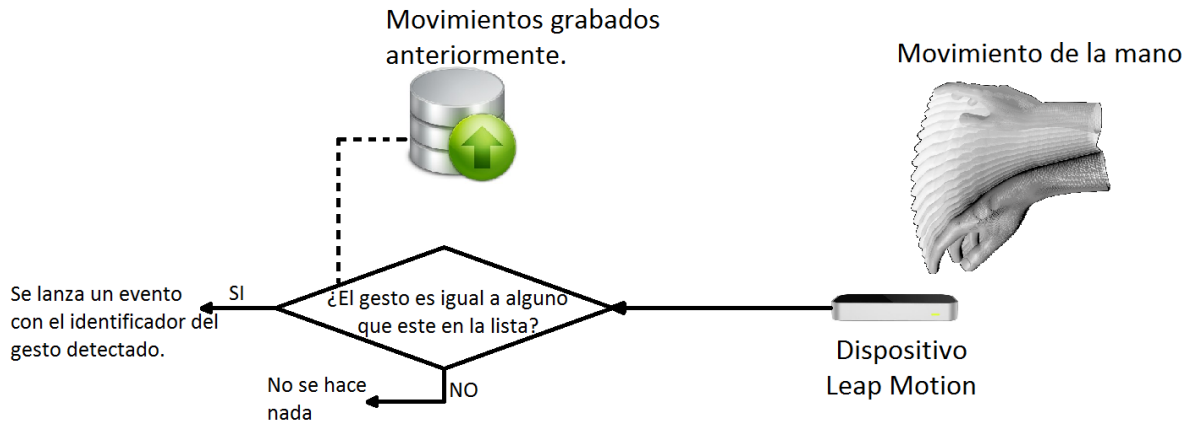

Figura 9 Funcionamiento del reconocedor [5]

Este proyecto está dividido en varias secciones:

1. Grabador de gestos: Permite al usuario grabar los gestos que crea necesarios para asociarlos a las acciones que crea oportunas. Esta función graba los gestos y guarda la información en un archivo.
    a. El usuario marca los parámetros que crea necesarios para definir el gesto.
    b. Graba el gesto correspondiente.
    c. La aplicación calcula la velocidad media del gesto si es necesaria.
2. Reconocimiento de gestos: A partir de los gestos grabados los compara con los recibidos por el dispositivo mientras que la aplicación está en funcionamiento para ver si coinciden y debe realizarse alguna acción o no.
    a. Algoritmo DTW [6]: Este algoritmo recibe dos secuencias para compararlas y deducir el grado de similitud entre ellas. Al trabajar con varias dimensiones, se empleará para la comparación de gestos una implementación de este algoritmo

llamada NDTW [7], que realiza esta comparación en varias dimensiones. Para nuestro caso las secuencias que recibirá son la secuencia formada por las listas de las posiciones X,Y,Z de la mano recibida por el dispositivo y otra secuencia con las posiciones del gesto grabado.

b. Buffer circular: En este buffer se irán introduciendo los *frames* recibidos por el dispositivo para su posterior comparación con los gestos almacenados por el grabador de gestos. Como no se puede saber en qué momento puede empezar a realizar un gesto se debe emplear un buffer circular para realizar la comparación cada vez que nos entra un nuevo *frame*, incorporando el nuevo al buffer y eliminando el más antiguo almacenado.

c. Estructura del reconocedor: Ya que el reconocedor tiene que realizar dos funciones distintas, se ha dividido en dos *threads* (cada uno ejecutando una funcionalidad) para un correcto funcionamiento del reconocedor: un *thread* escuchador que va introduciendo al buffer circular los *frames* que se obtienen del dispositivo Leap Motion y un *thread* comparador que es llamado cada vez que se inserta un nuevo *frame* en el buffer circular para comparar el contenido del buffer con los gestos previamente grabados.

3. ESTADO DEL ARTE

3.1. Lenguajes de programación

3.1.1. C#:

C# es un lenguaje de programación que se ha diseñado para compilar diversas aplicaciones que se ejecutan con el Framework .NET. C# es simple, eficaz, con seguridad de tipos y orientado a objetos. Para la implementación de C# en este proyecto se ha empleado la herramienta Visual Studio 2010 Professional. Visual C# es una implementación del lenguaje C# de Microsoft, Visual Studio ofrece compatibilidad con Visual C# con un completo editor de código, un compilador, plantillas de proyecto, diseñadores, asistentes para código, un depurador eficaz y de fácil uso y otras herramientas. La biblioteca de clases de .NET Framework ofrece acceso a numerosos servicios de sistema operativo y a otras clases útiles y adecuadamente diseñadas que aceleran el ciclo de desarrollo de manera significativa.

3.1.2. ActionScript:

ActionScript es el lenguaje de programación de la plataforma Adobe Flash. Originalmente fue desarrollado como una forma para que los desarrolladores puedan programar de una forma más interactiva. La programación con ActionScript permite mucha más eficiencia en las aplicaciones de la plataforma Flash para construir animaciones de todo tipo, desde simples a complejas, ricas en datos e interfaces interactivas. Para este lenguaje de programación se han empleado dos herramientas: Adobe Flash Builder 4.6 y Adobe Flash Profesional CS3.

3.2. SDK

Una de las tareas del trabajo de fin de grado es actualizar el reconocedor a la última versión disponible del SDK ya que se han implementado grandes mejoras en reconocimiento y rendimiento del dispositivo Leap Motion. De manera que se ha pasado de emplear el SDK 1.6 al 2.2.2.24469 [8] que es la última versión disponible durante el desarrollo del proyecto. Por eso he investigado todas las novedades que ofrecen los nuevos SDK disponibles en el foro de desarrolladores [9] donde se encuentran disponibles.

Entre las múltiples novedades la primera que hay que destacar es la versión 2.1.2.21921 [10] donde se consiguió que las aplicaciones de V1 sean compatibles con el nuevo seguimiento de la V2. Dado el objetivo de actualizar el reconocedor de Leap Motion del SDK1.6 al último disponible de la V2 esta compatibilidad ha facilitado el trabajo.

Respecto al esqueleto de la mano que detecta el dispositivo se ha obtenido un nuevo punto de seguimiento que es la muñeca. Al obtener esta articulación el reconocedor genera un esqueleto más humano gracias a la detección del ángulo que forma el brazo con la mano, lo cual reduce en gran medida los problemas surgidos de mostrar el anverso de la mano al dispositivo (Figura 10), lo que hacía que el reconocedor marcara el anverso de la mano izquierda como una mano derecha.

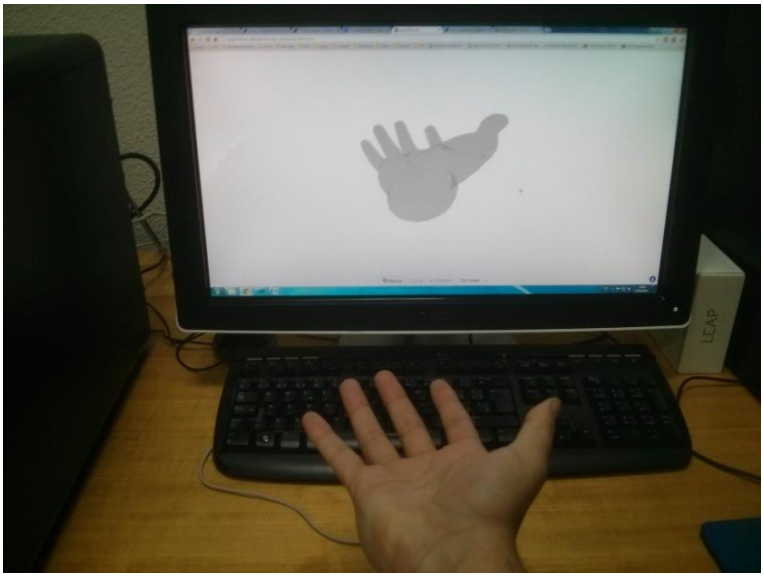

Figura 10 Anverso de la mano

Para el seguimiento de la mano se han conseguido grandes mejoras de posicionamiento. Durante el SDK1 en cuanto un elemento se encontraba fuera del área de detección o en un punto ciego de las cámaras el dispositivo daba posiciones relativas de ese elemento ilógicas.

Como ejemplo usaremos una mano de canto (Figura 11) en la que el dedo índice se encuentra en un punto ciego de la cámara debido a que los otros dedos están en medio, con el SDK 2 lo posiciona en el lugar que más probabilidad tenga de localizarse (que es encontrarse extendido tras los otros dedos). Esto también se aplica a otros casos como serían cerrar la mano y voltearla, el reconocedor supone que los dedos siguen flexionados tal y como estaban mientras los podía detectar.

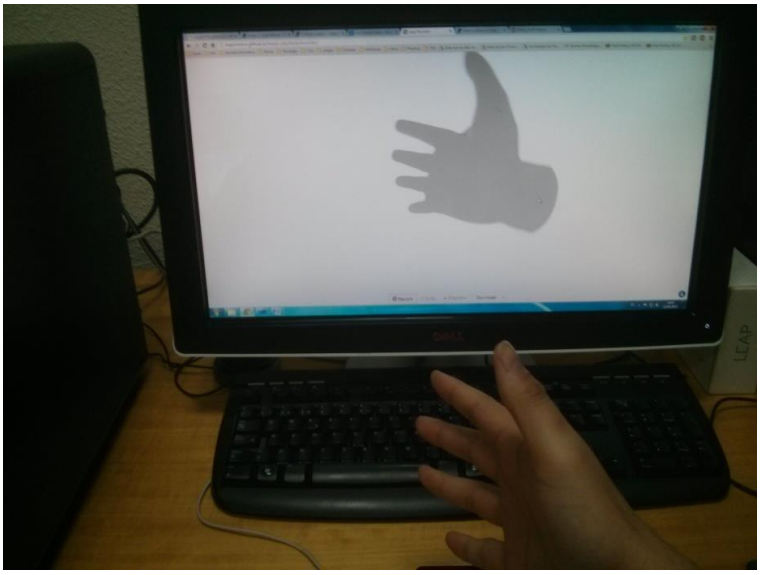

Figura 11 Canto de la mano

Se han logrado grandes mejoras en el reconocimiento de gestos circulares. Esto se debe a que ahora se puede obtener el centro del giro para el brazo y la muñeca. Con estos centros y el de la palma de la mano puede obtener una información mucho más fiable que la que obtenía antes con solo la palma ya que la posición de los dedos no ofrecía la suficiente confianza debido a que pueden tener movimientos extra.

También se ha modificado el sistema de toma de imágenes y la forma de acceder a él. El dispositivo obtiene imágenes con más calidad que antes (Figura 12). También se ha realizado grandes mejoras de rendimiento que han permitido subir la tasa de *frames* por segundo que captura el dispositivo. Anteriormente se accedía a las imágenes de los *frames* mediante un método, ahora las han hecho formar parte de las propiedades del *frame* reduciendo el tiempo de acceso a la imagen.

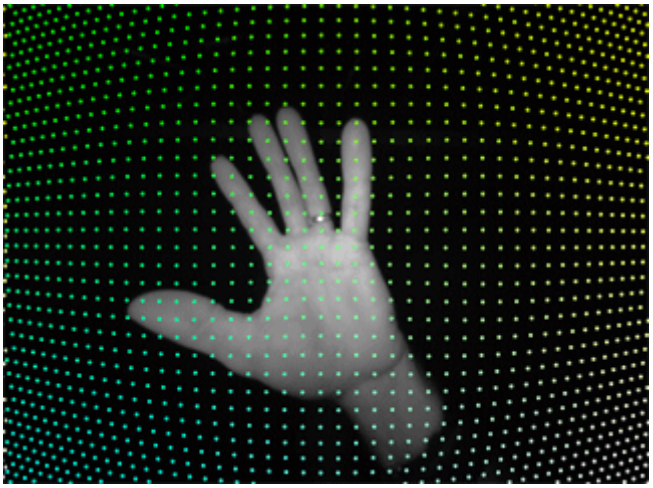

Figura 12 Imagen RAW de una cámara [11]

Entre las novedades del cambio de SDK, los desarrolladores también nos han informado de problemas que han encontrado y que están trabajando para arreglar para las próximas versiones:

- Debido al cambio de la toma de imágenes todas las aplicaciones desarrolladas anteriores al SDK 2.2.2 deben de cambiar todas las llamadas al método *Controller.Images()* por la propiedad *Controller.Images*.
- Las aplicaciones construidas con los SDK 2.1.0 o 2.1.1 deben actualizar los DLLs a los de la versión 2.2.0.
- *Avast! Antivirus* puede interferir con la instalación del servicio de Leap Motion, por lo que se recomienda desactivar sus escudos para que la instalación sea correcta.
- Las funciones *sphereRadius* y *sphereCenter* de la clase *Hand*, que obtienen información sobre movimientos circulares, se vuelven inestables cuando la mano se encuentra totalmente abierta.
- La calidad del seguimiento se ve reducida cuando se realiza con un puño cerrado o con un solo dedo extendido.
- El seguimiento podría no funcionar con la suficiente eficiencia si el usuario lleva pulseras, anillos, relojes, mangas largas, etc.

3.3. Reconocedor de Leap Motion

Mediante el estudio realizado para la actualización del nuevo SDK de Leap Motion y una vez confirmadas sus mejoras hay que actualizar el prototipo básico que hay disponible para la grabación y el reconocimiento con Leap Motion. Para ello hay que estudiar varios elementos del proyecto:

3.3.1. Esqueleto

Un esqueleto es la captura de un instante como una imagen. Tener una lista de esqueletos es como tener varias imágenes que se tomaron sucesivamente en un instante de tiempo. Si se muestra la lista de esqueletos en sucesión sería como ver un vídeo en el que se mueven una o varias manos. Cada esqueleto está formado por un conjunto de Joint.

3.3.1.1. Joint

El Joint está formado por el nombre de un miembro, un atributo que define si está involucrado en el movimiento y su posición en el eje X, Y, Z. El nombre tiene que tener un valor determinado y contiene la información sobre las coordenadas que le corresponden:

- Mano derecha
- Mano izquierda
- Brazo derecho
- Brazo izquierdo

Debido a los gestos que se emplearán en el juego con estos Joints serán suficientes para definirlos correctamente. También podrían generarse 10 Joints más para guardar las posiciones que tienen cada uno de los 10 dedos, pero al no ser necesarios en esta aplicación generan mucha información extra que ralentiza la velocidad para reconocer los gestos.

3.3.2. Definición de gestos

Para la realización de los mini juegos que tiene Interacttico se requieren definir seis gestos: izquierda, derecha, seleccionar, bajar, abrir y rotar. Hay que intentar que los gestos sean definidos de tal forma que resulten intuitivos y puedan realizarse sin necesidad de ser explicados. Para facilitar la comprensión sobre qué dirección lleva el gesto se incluye una imagen (Figura 13) con los ejes de coordenadas que ofrece Leap Motion.

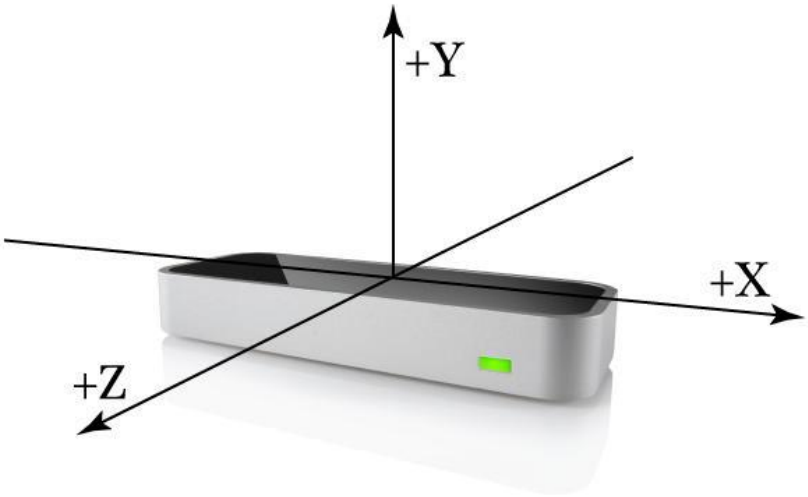

Figura 13 Sistema de coordenadas de Leap Motion [12]

3.3.2.1. Izquierda

El gesto se empleará durante la selección de un elemento de la nube moviendo el cuadro de selección a la izquierda. Para la realización de este gesto se comprobará que en el eje de coordenadas X se produzca un decremento (Figura 14).

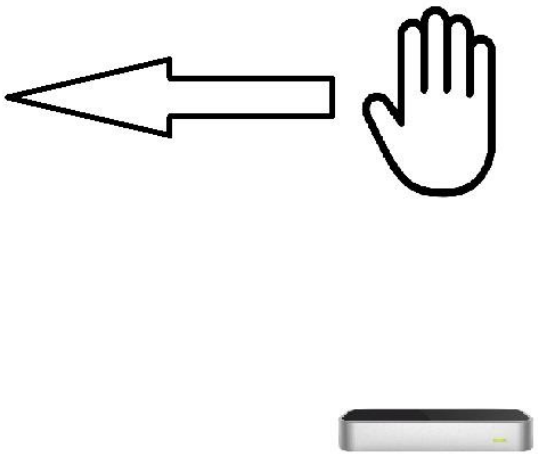

Figura 14 Gesto Izquierda

3.3.2.2. Derecha/Lanzar

El gesto se empleará para dos funcionalidades distintas: lanzar el elemento que hay en la cinta hacia la caja o durante la selección de un elemento de la nube moviendo el cuadro de selección a la derecha. Para la realización de este gesto se comprobará que en el eje de coordenadas X se produzca un incremento (Figura 15).

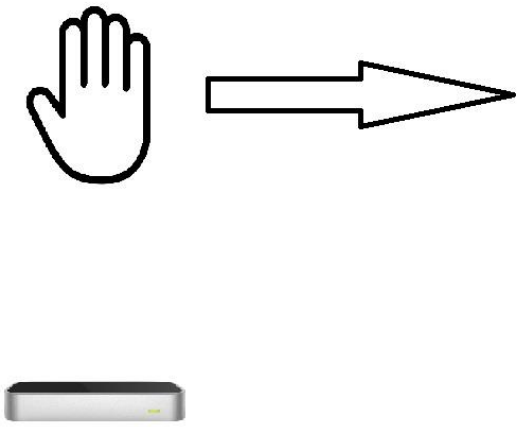

Figura 15 Gesto Derecha/Lanzar

3.3.2.3. Seleccionar

El gesto se emplea para seleccionar un elemento que puede ser un elemento de la nube durante la fase de selección o durante el mini juego de asociar conceptos para seleccionar la parte que corresponda de la caja donde habrá que introducir el elemento de la cinta. Para la realización de este gesto se comprobará que en el eje de coordenadas Z se produzca un decremento (Figura 16).

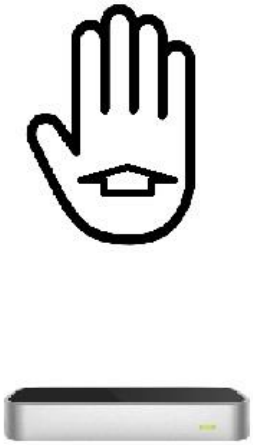

Figura 16 Gesto Seleccionar

3.3.2.4. Bajar

El gesto se emplea para bajar el elemento que ha sido seleccionado correctamente desde la nube hacia la cinta. Para la realización de este gesto se comprobará que en el eje de coordenadas Y se produzca un decremento (Figura 17).

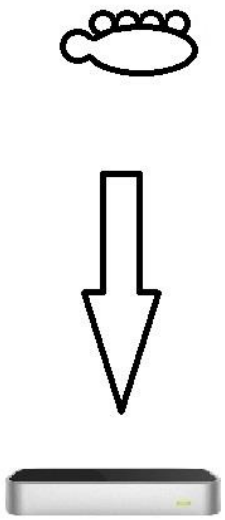

Figura 17 Gesto Bajar

3.3.2.5. Abrir

El gesto se emplea para abrir el cofre del juego DELETREA o en los juegos de agrupar conceptos para abrir la caja donde se depositará el elemento que debe ser seleccionado. Para la realización de este gesto se comprobará que en el eje de coordenadas X se incremente la distancia entre ambas manos, es decir, separarlas entre sí (Figura 18).

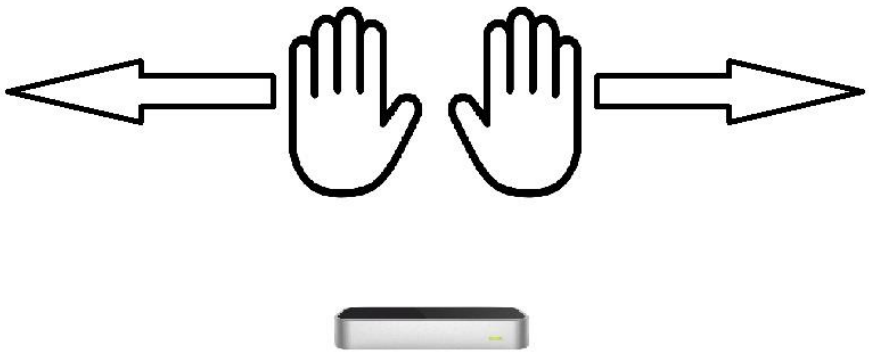

Figura 18 Gesto Abrir

3.3.2.6. Rotar

El gesto se emplea solo en el desarrollo del juego DIFICIL ya que es el único en el que el usuario debe realizar un gesto que haga rotar la caja. Para la realización de este gesto se comprobará que la mano hace un semicírculo en el sentido de las agujas del reloj (Figura 19).

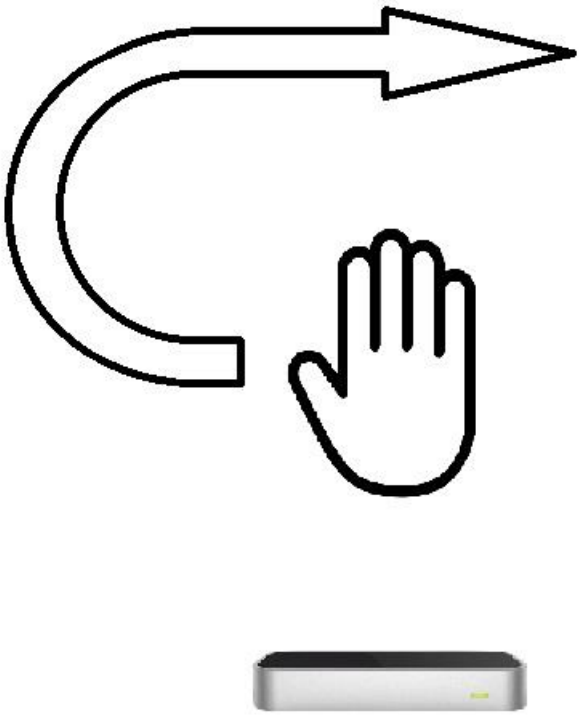

Figura 19 Gesto Rotar

3.3.3. Reconocimiento de gestos

Una de las partes más complejas del proyecto es el reconocimiento de gestos. La interacción con Leap Motion nos devuelve información sobre las coordenadas de los distintos elementos que componen las manos que son los dedos, las palmas y las muñecas para cada momento capturado (*frame*). Mediante la comparación de estas coordenadas con las de los gestos predefinidos se obtiene el gesto que ha realizado el usuario.

Para poder realizar este reconocimiento se ha partido del prototipo básico realizado por Karim Laazizi en junio de 2014. Para entender el funcionamiento de este reconocedor hay que tratar las distintas partes que lo componen.

3.3.3.1. Algoritmo NDtw

Para la comparación de los *frames* obtenidos con los gestos predefinidos se empleará el algoritmo NDtw. Dicho procedimiento recibe como parámetros dos secuencias que analizará para determinar el grado de similitud. Para el caso de Leap Motion esos argumentos serán una lista con las coordenadas X, Y, Z de los *frames* obtenidos y otra lista con las coordenadas ya guardadas de cada gesto que hemos grabado previamente para comparar ambas.

3.3.3.1.1. Algoritmo DTW

El algoritmo DTW, en español llamado "Alineamiento Temporal Dinámico", sirve para medir la similitud entre dos secuencias temporales que pueden variar en el tiempo o la velocidad. En general, este algoritmo calcula una coincidencia óptima entre dos secuencias dadas. Estas secuencias no tienen por qué tener la misma longitud. Para entenderlo mejor usaremos un ejemplo sencillo [6]. Tenemos 2 secuencias A y B (Figura 20):

A = a1, a2, a3,..., ai,..., an.
B = b1, b2, b3,..., bi,..., bm.

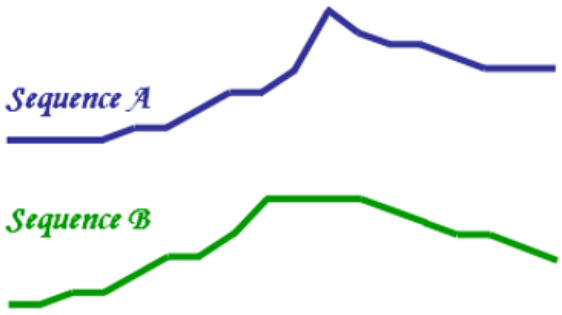

Figura 20 Secuencias

Estas dos secuencias se dibujaran en los lados de una cuadrícula, uno en la parte superior, y otro en la parte izquierda. Dentro de cada celda se coloca la diferencia comparando los elementos correspondientes de las dos secuencias. Para encontrar la mejor coincidencia o alineación entre estas dos secuencias se tiene que encontrar un camino a través de la cuadrícula que minimiza la distancia total entre ellos. En la Figura 21 podemos ver el camino mínimo en forma de puntos rojos de la secuencia A y B.

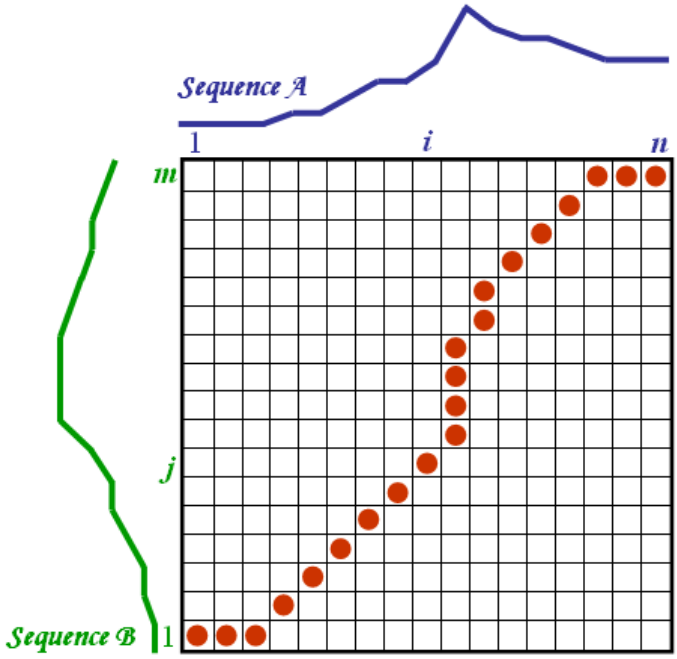

Figura 21 Ejemplo al aplicar el algoritmo DTW [6]

Siguiendo el ejemplo, con las secuencias A=(2,3,5,7,10,6,3) y B=(1,2,5,8,9,7,2), el cálculo de las diferencias quedaría (Figura 22):

X	2	3	5	7	10	6	3
2	0	1	3	5	8	4	**1**
7	5	4	2	0	3	**1**	4
9	7	6	4	2	**1**	3	6
8	6	5	3	**1**	2	2	5
5	3	2	**0**	2	5	1	2
2	0	**1**	3	5	8	4	1
1	**1**	2	4	6	9	5	2

Figura 22 Cálculo de las diferencias con el algoritmo DTW [5]

El procedimiento para el cálculo de esta distancia en general consiste en encontrar todas las rutas posibles a través de dicha red y para cada una calcular la distancia total. En el ejemplo de la figura anterior, el camino optimo seria: 1+1+0+1+1+1+1 = 6.

Dado que un análisis en profundidad para crear una implementación de dicho algoritmo conllevaría mucho trabajo (aproximadamente el tiempo de un Trabajo de Fin de Grado), se empleó una implementación ya hecha del algoritmo en C# con licencia MIT, que permite usarlo sin restricciones. La implementación empleada es NDtw [7], que consta de 2 paquetes importantes:

- NDtw: El algoritmo.
- NDtw.Visualization.WPF: Visualización grafica de los resultados (Figura 23).

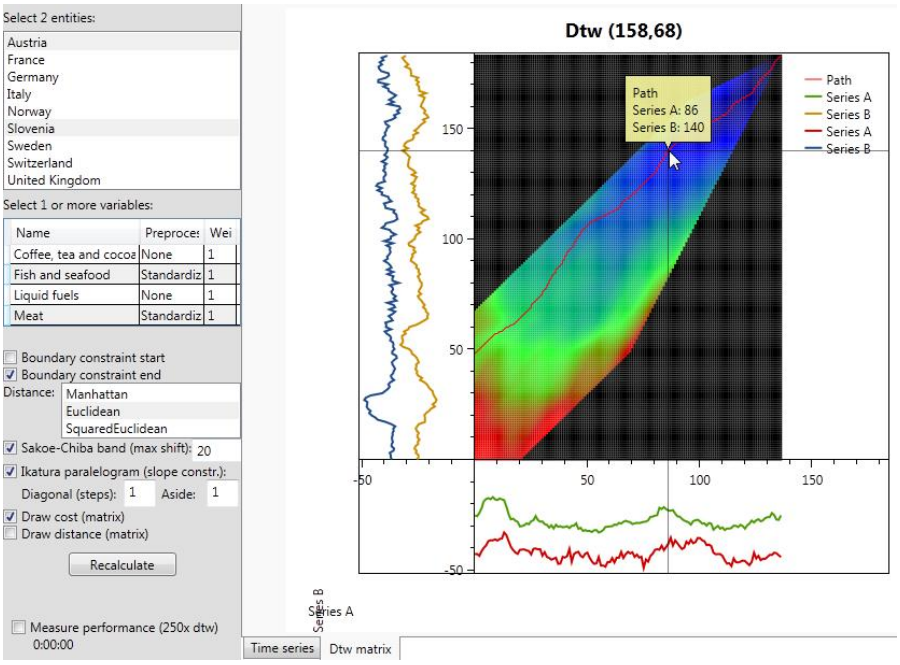

Figura 23 Visualización gráfica de los resultados NDtw

Esta implementación es fácil de usar, ya que solo se tiene que llamar al método y pasarle las dos secuencias que se quieren comparar. Además permite modificar diferentes parámetros al inicializarlo para hacerlo más eficiente.

3.3.3.2. Buffer circular

Debido a que no sabemos en qué momento exacto se comenzará la realización de un gesto se empleará un buffer circular (Figura 24) para el almacenamiento temporal de los *frames* obtenidos durante la ejecución. De esta manera podremos comparar el gesto realizado con los predefinidos.

En cada elemento del buffer hay información sobre el *frame*:

- Posición X, Y, Z de la mano derecha.
- Posición X, Y, Z de la mano izquierda.
- Posición X, Y, Z del brazo derecho.
- Posición X, Y, Z del brazo izquierdo.
- Lista con las posiciones X, Y, Z de los dedos de la mano derecha.
- Lista con las posiciones X, Y, Z de los dedos de la mano izquierda.
- La mano empleada en el gesto: derecha, izquierda o ambas.

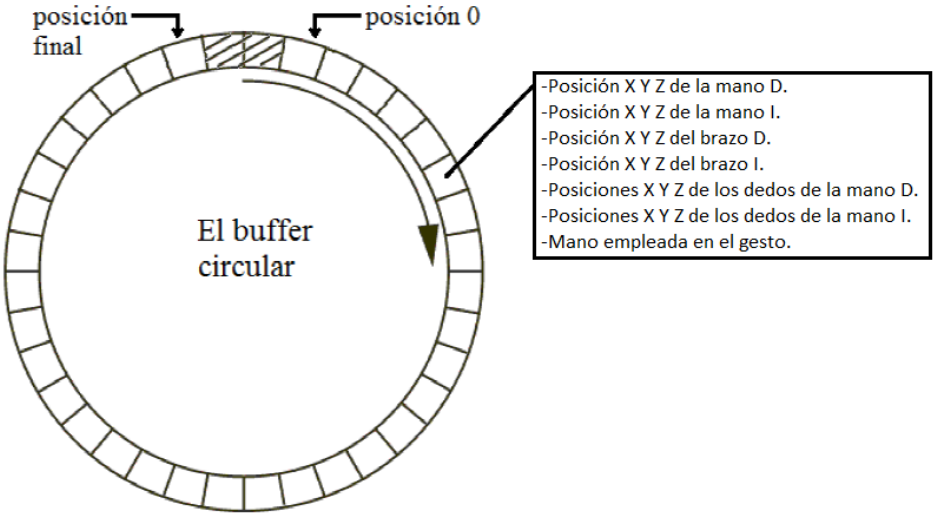

Figura 24 Buffer circular

3.3.3.3. Estructura del reconocimiento.

Para el correcto funcionamiento del reconocimiento se ha dividido en dos hilos:

- Escuchador: es la parte que recibe los datos de Leap Motion y los almacena en el buffer circular. Después de añadir el nuevo dato al buffer, si el buffer tiene al menos 25 *frames* lo clonará y lo incluirá a la cola de buffers. Esta cola se emplea para no sobrescribir los datos del buffer y no perder información. Finalmente envía un evento al comparador para que sepa que tiene buffers esperando ser analizados.

- Comparador: se encargará de recoger los buffer clonados cuando los haya, obtener la información contenida en cada buffer y comparar sus datos con los de los gestos previamente almacenados.

Para facilitar el entendimiento del reconocimiento se incluye un esquema que lo resume (Figura 25).

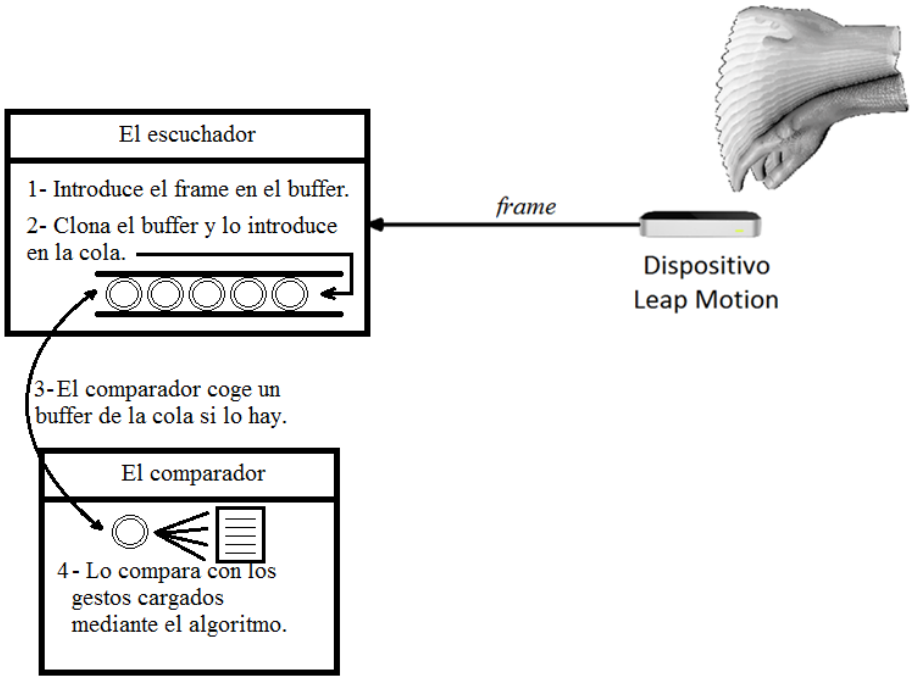

Figura 25 Esquema del reconocimiento de gestos [5]

3.4. Definición de la información necesaria para el estudio de la interacción con Leap Motion

Para facilitar el estudio de la interacción se ha añadido mucha más información sobre los *frames* que se obtienen durante la ejecución del juego que la grabada para comparar gestos, ya que para realizar la comparación de los gestos no es necesaria excesiva información pero puede resultar de utilidad para la realización de este estudio.

Hay que definir qué información puede resultar útil para su posterior estudio, para ello primero se debe ver los datos que se almacenan en la base de datos sobre los otros dispositivos:

- Teclado: el tiempo máximo que se ha estado pulsando cada tecla y el número de repeticiones seguidas que se pulsa la misma tecla.
- Pantalla táctil: la precisión mayor, menor y media con la que se realiza un gesto, es decir, el parecido entre el gesto realizado y el gesto esperado mediante la comparación de las coordenadas.
- Kinect: el número de repeticiones que se hace cada gesto durante la sesión.
- Errores: para todos los dispositivos se guarda cuándo se ha realizado un gesto incorrecto, el gesto realizado y el gesto esperado.

Después de revisar la información que el juego guarda en la base de datos tras finalizar cada sesión se define qué datos podemos almacenar sobre Leap Motion. Los errores seguirán almacenándose automáticamente cada vez que se realicen sin importar el dispositivo que se haya empleado, así que solo hay que centrarse en los datos sobre el dispositivo Leap Motion:

- Mano empleada: uno de los datos que resultan de mayor interés es qué mano suele emplear el usuario para realizar los gestos durante la ejecución del juego. Para facilitar su recogida se guarda en el buffer circular la información sobre qué mano se está empleando. En este caso la información sería una cadena de texto que puede ser “izquierda”, “derecha” o “ambas”.
- Velocidad del gesto: el reconocedor permite calcular la velocidad con la que se realiza un gesto, por lo que puede resultar de interés almacenar cuál es la velocidad media con la que se realiza cada gesto durante la sesión. De esta forma se puede ver las capacidades motrices que tienen los usuarios viendo la velocidad con la que realizan gestos.

- Repeticiones: otro dato que puede ser interesante es hacerlo similar a Kinect recogiendo cuántas veces se repite cada gesto que se realiza durante la sesión mediante el uso de contadores, por ejemplo, durante la realización del juego se ha realizado el gesto abrir siete veces. Se puede también aprovechar la mano empleada para obtener el número de veces que se repite cada gesto con cada mano:
    - gesto izquierda: empleando la mano izquierda
    - gesto izquierda: empleando la mano derecha
    - gesto derecha: empleando la mano izquierda
    - gesto derecha: empleando la mano derecha
    - gesto seleccionar: empleando la mano izquierda
    - gesto seleccionar: empleando la mano derecha
    - gesto bajar: empleando la mano izquierda
    - gesto bajar: empleando la mano derecha
    - gesto rotar: empleando la mano izquierda
    - gesto rotar: empleando la mano derecha
    - gesto abrir: empleando ambas manos
- Precisión media del gesto: Al estar trabajando con coordenadas se puede definir una precisión al igual que se hace con el dispositivo pantalla táctil. Para adaptarlo en este caso podemos trabajar directamente sobre el algoritmo NDtw que es el encargado de medir la similitud mediante el cálculo de costes.

  En lugar de calcular la precisión empleando la diferencia entre la coordenada en la que selecciona el usuario y la coordenada esperada se puede calcular empleando los costes obtenidos por el algoritmo. El coste mínimo para que un gesto sea reconocible es de 1200, todo por debajo de esa cantidad no se reconocerá como un gesto, por lo que se puede elegir como límite al que se le asignaría un valor de 0% e irlo incrementando según aumente coste.
- Precisión por dimensión: Al tener múltiples costes, uno para cada dimensión, se puede también guardar la precisión con la que se realiza cada gesto para cada dimensión. Se realizaría un cálculo como el anterior según el coste obtenido para cada dimensión. Para realizar este caso habría que revisar los costes mínimos necesarios para cada dimensión, revisar qué dimensiones se tienen en cuenta para la realización de cada gesto y qué dimensión es más importante para el reconocimiento del gesto.

4. DESAROLLO

Para el desarrollo del proyecto se ha actualizado el grabador y reconocedor de gestos de Leap Motion, el Wrapper que comunica al juego con los dispositivos, la lógica del juego y el cargador flash con las imágenes de los distintos mini juegos. En la Figura 26 se ven las diferentes partes del proyecto y se marcan en azul las que han sido modificadas.

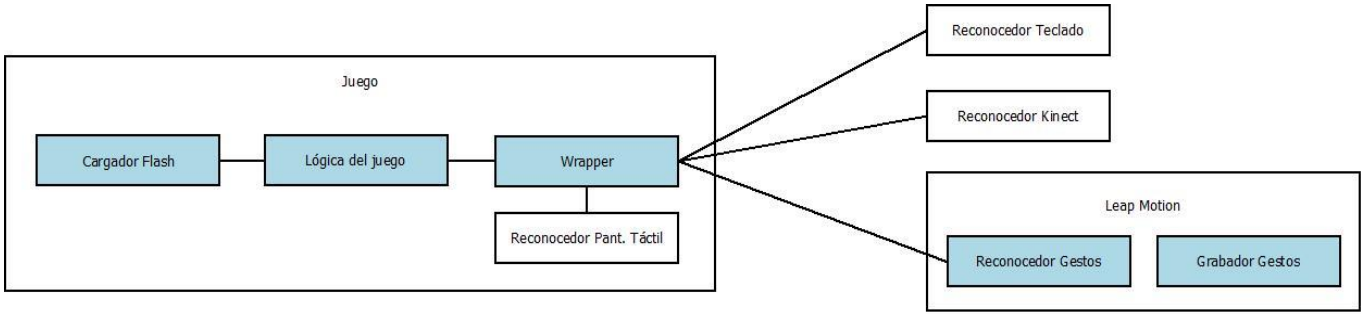

Figura 26 Esquema del Proyecto

4.1. Requisitos del sistema

Para el desarrollo de este proyecto se deben tener en cuenta dos tipos de requisitos: los requisitos sobre el reconocedor de gestos de Leap Motion y los requisitos sobre los gestos que deben ser definidos para el correcto funcionamiento del juego.

4.1.1. Requisitos del reconocedor Leap Motion

Del prototipo de Leap Motion del que partimos emplearemos el grabador para definir los gestos que emplearemos en el juego y se tendrá que extraer el reconocedor para introducirlo en el juego tras actualizarlo con el nuevo SDK y aumentar la información que recoge. Los requisitos de este proyecto son:

- Permitir definir gestos: el usuario debe poder grabar un movimiento con su mano para definirlo como un gesto.
- Reconocer un gesto: se debe reconocer que el movimiento que se está realizando es similar a uno ya grabado.
- Velocidad del gesto: al crear un gesto el usuario decide si se debe tener en cuenta la velocidad con la que se realiza el gesto o no es necesario. Si gesto tiene definido que es necesario tener en cuenta la velocidad el grabador calculará la velocidad con la que se graba el gesto y la almacenará en su esqueleto y el reconocedor calculará la velocidad del gesto realizado y la comparará con la del gesto grabado.

- Eje del movimiento: al crear un gesto el usuario decide qué ejes (Figura 13) deben tenerse en cuenta para la realización del gesto, ya que hay gestos (izquierda por ejemplo) que no necesitan tener en cuenta todos los ejes. Como con la velocidad se tendrán en cuenta los ejes para reconocer la similitud de un gesto.
- Gestor de gestos: al grabar un gesto se puede añadir a un archivo nuevo o uno con más gestos guardados. Para poder reconocer gestos debe permitir cargar un archivo que almacene gestos.
- Interfaz gráfica: para facilitar el trabajo al usuario debe tener una interfaz gráfica sencilla para que el usuario sepa qué está haciendo en todo momento.
- Configuración del reconocedor: el usuario debe poder modificar los valores por defecto del reconocedor: el coste para el reconocimiento, *frame* de inicio, *frame* de fin y número de *frames* de un gesto entre otros.

4.1.2. Requisitos de los gestos

Para el correcto funcionamiento del juego se necesita una batería de gestos predefinidos que se deben grabar para que el reconocedor pueda compararlos y reconocerlos correctamente permitiendo la continuación de la ejecución del juego. Al tener múltiples mini juegos que son similares en características se toma el juego que tiene mayor complejidad en cuanto a variedad de gestos que se deben emplear que es el de agrupar conceptos en el nivel de dificultad difícil donde obtenemos las siguientes fases:

1. Rotar la caja: inicialmente hay que rotar la caja con los distintos grupos con los que hay que hacer coincidir los elementos de la nube. De esta fase se deduce que es necesario un gesto para “rotar”.
2. Detener la rotación: una vez que la caja está rotando el usuario debe pararla en el apartado que el juego le haya pedido cuando la flecha lo esté apuntando mientras rota, por tanto necesitamos un gesto “seleccionar” para elegir el elemento.
3. Abrir la caja: una vez detenida la caja en el apartado correcto hay que abrirla para que se pueda introducir en su interior el elemento necesario, lo que implica la necesidad de un gesto para “abrir”.
4. Selección en la nube: en la nube superior hay múltiples elementos para seleccionar, se necesita poder moverse entre ellos y seleccionar, por lo que obtenemos tres gestos nuevos: “izquierda”, “derecha” y “seleccionar” (ya definido).

5. Bajar a la cinta: una vez seleccionado de la nube el elemento correcto debe bajarse a la cinta para poder introducirlo posteriormente a la caja. Se genera un nuevo gesto "bajar".
6. Lanzar a la caja: una vez se encuentra el elemento en la cinta hay que introducirlo en la caja de la derecha, por lo que hay que empujarlo o lanzarlo hacia la derecha, para ello se emplea el gesto "derecha" ya que es idéntico al no tener en cuenta la fuerza con la que se realiza el gesto.
7. Finalizar: tras introducirse el elemento en la caja se finaliza la iteración. Si aún quedan elementos por asignar en la nube se vuelve al primer paso y en caso de no quedar ninguno se finaliza la sesión, mostrando la puntuación obtenida y almacenando los datos de la sesión en la base de datos.

Tras el estudio de los gestos necesarios para el correcto desarrollo del juego obtenemos seis gestos necesarios: Rotar, Seleccionar, Abrir, Izquierda, Derecha y Bajar.

4.2. División del proyecto

Una de las dificultades a las que hay que enfrentarse para desarrollar este proyecto es que el código está realizado en múltiples lenguajes. Para facilitar su comprensión se explicará para qué se ha empleado cada lenguaje y las herramientas de desarrollo con las que se ha desarrollado. En este proyecto se han empleado dos lenguajes de programación distintos: ActionScript y C#. Para facilitar al lector el entendimiento del sistema se adjunta un diagrama de clases con las partes que conforman este proyecto (Figura 27).

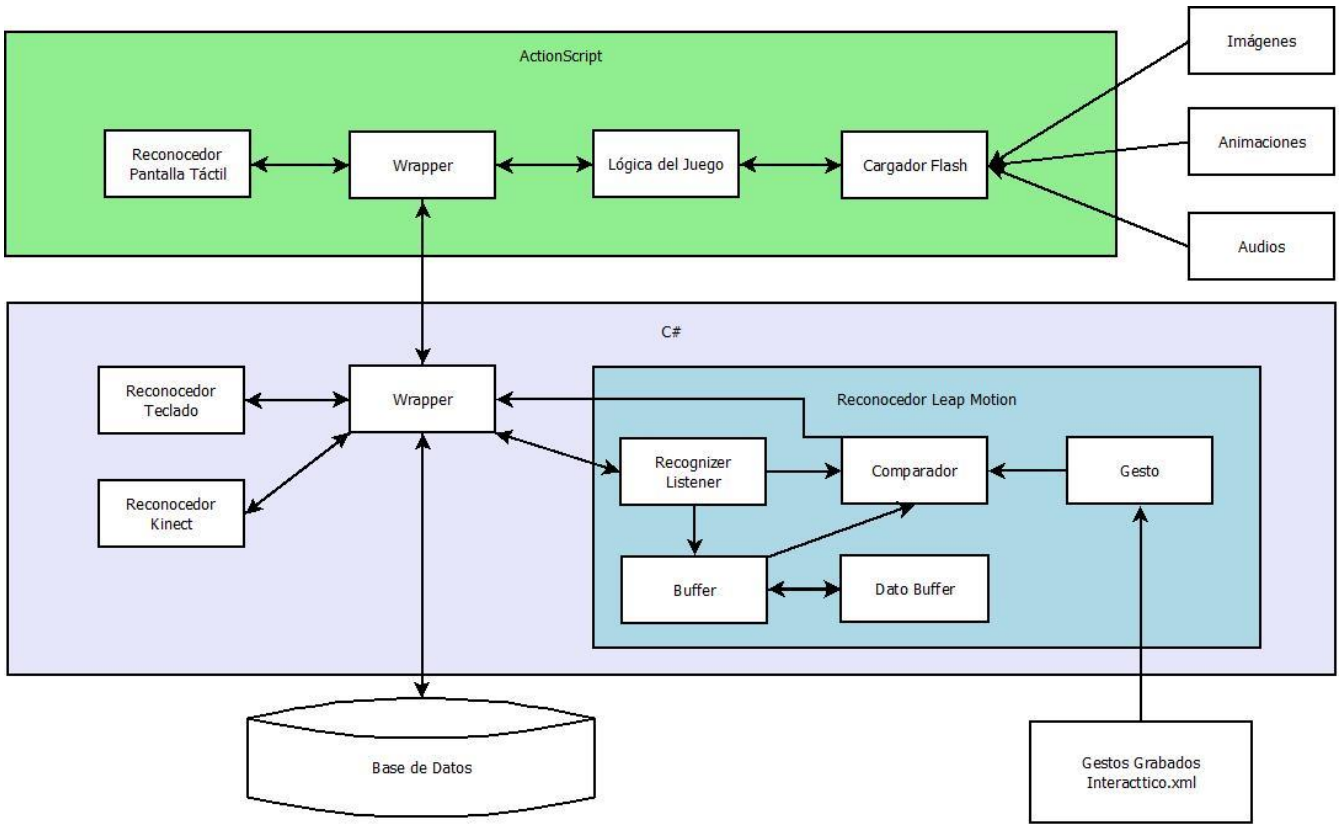

Figura 27 Diagrama de clases

4.2.1. C#

En este proyecto se ha empleado C# para implementar varios de los reconocedores que empleamos: el reconocedor del teclado, de Kinect, de Leap Motion y un *wrapper* para la comunicación entre los reconocedores y la lógica del juego realizada con ActionScript. Es importante de cara a la realización de este Trabajo de Fin de Grado ya que se ha empleado mucho tiempo para mejorar el reconocedor de Leap Motion. También se ha actualizado el *wrapper* de C# para comprobar si hay algún dispositivo de Leap Motion disponible al iniciar el juego para permitir seleccionarlo, activar su reconocedor cuando se selecciona y desactivarlo cuando se finaliza la sesión.

4.2.2. ActionScript

La herramienta Flash Builder 4.6 se ha empleado para la implementación de la lógica del juego, el reconocedor de la pantalla táctil y un *wrapper* en ActionScript para comunicarse con los reconocedores programados en C#. En este Trabajo de Fin de Grado ha sido necesario para actualizar la lógica del juego, donde se ha tenido que determinar qué acciones se pueden realizar durante las distintas fases de los mini juegos, notificar y registrar en la base de datos los errores cometidos y comunicarse con los elementos flash para que realicen las animaciones correspondientes. También se ha actualizado el *wrapper* para poder recibir las acciones que han sido reconocidas con Leap Motion, la petición para activar Leap Motion cuando ha sido seleccionada por el usuario y la petición para desactivarlo cuando se finaliza la sesión.

También se ha empleado la herramienta Adobe Flash Professional CS3 para la creación de las animaciones y las acciones que las activan. Durante el desarrollo de este Trabajo de Fin de Grado se ha utilizado para generar un botón que permita al usuario emplear el dispositivo Leap Motion, debe desactivarse en caso de no estar el dispositivo disponible, mandar un mensaje para activar el reconocedor cuando lo seleccione el usuario y mandar otro para desactivarlo cuando finalice la sesión.

4.2.3. Wrapper

Para poder realizar las comunicaciones entre ambos lenguajes se han definido dos clases llamadas Wrapper, cada una de ellas se realiza en uno de los lenguajes. Ambos *wrappers* se comunican entre ellos mediante el envío de eventos. El *wrapper* de C# comprueba qué dispositivos tiene disponibles al iniciar el juego comunicándoselo al otro *wrapper* para que la lógica trabaje en base a ello. El *wrapper* de C# recibe los gestos obtenidos por los reconocedores de teclado, Kinect y Leap Motion en caso de ser seleccionado y envía un evento con el gesto empleado al otro *wrapper* para que se lo comunique a la lógica del juego y ser evaluado. Tras ser evaluado se realizan las acciones oportunas y el *wrapper* de ActionScript envía la respuesta al otro *wrapper* para que la información de la interacción pueda ser almacenada en la base de datos. Para la pantalla táctil se comunica directamente con el *wrapper* de ActionScript, evitando la necesidad de la comunicación entre ambos *wrappers*.

4.3. Actualización del Reconocedor

Tras el estudio de la actualización del SDK que puede emplear Leap Motion y una vez confirmadas sus mejoras hay que actualizar el prototipo (Figura 28) que hay disponible para la grabación y el reconocimiento con Leap Motion. Para ello hay que modificar las librerías que se emplean además de llevar a cabo varias tareas:

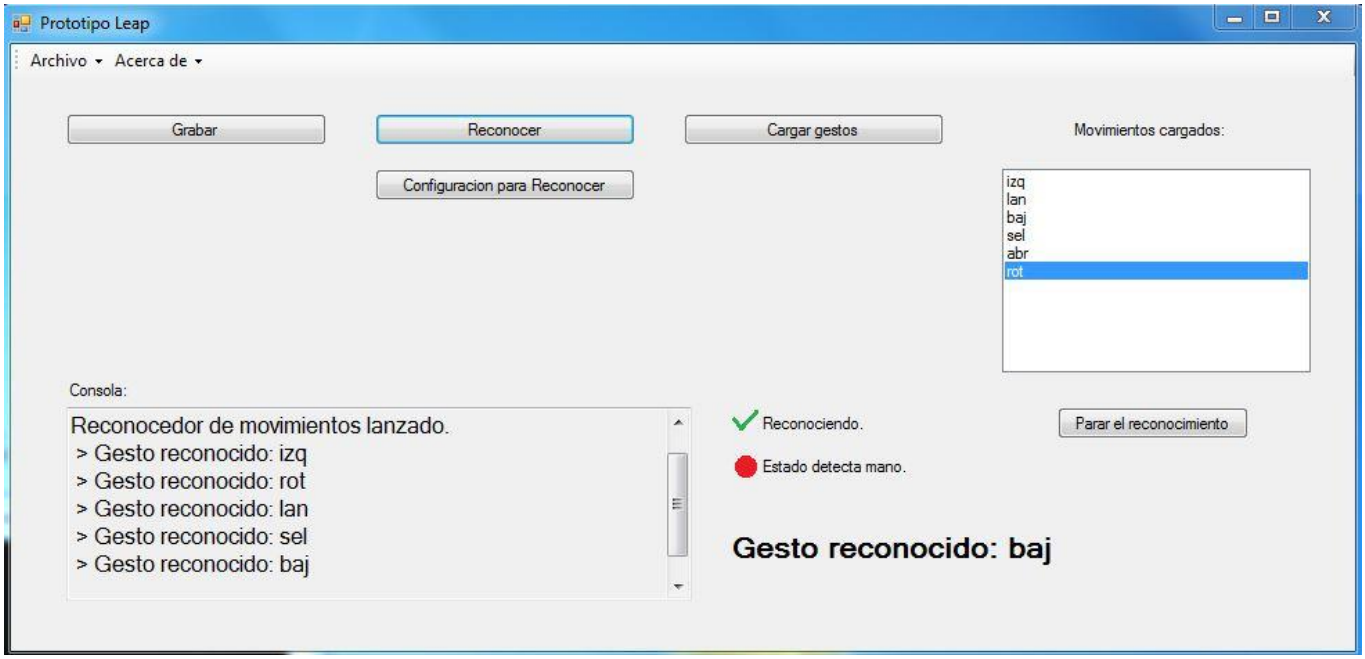

Figura 28 Prototipo Leap Motion

4.3.1. Esqueleto

El esqueleto inicial solo contenía la información de las coordenadas que tenía la palma de la mano. Para mejorar sus posibilidades con el nuevo SDK se le ha introducido mayor información aún. Los esqueletos pueden introducir tantos Joints como quisieran, facilitando la inclusión de estos nuevos elementos. Con los cambios introducidos en el programa se obtendrán dos tipos distintos de esqueletos:

4.3.1.1. Esqueleto grabado

Debido a la sencillez de los gestos que se emplean en el desarrollo del juego es suficiente con que el esqueleto contenga información sobre las posiciones X, Y, Z de las manos, también se obtienen para las muñecas ya que permiten entender mejor la posición de su mano. También los parámetros opcionales que ofrece el reconocedor como los ejes en los que hay que tener

en cuenta el reconocimiento o la velocidad con la que se debe realizar el gesto. De esta forma obtenemos:

- Nombre del gesto
- Identificador del gesto
- Dispositivo empleado
- Velocidad mínima
- Velocidad máxima
- Participa velocidad
- Participa X
- Participa Y
- Participa Z
- Posición X, Y, Z de la mano derecha
- Posición X, Y, Z de la mano izquierda
- Posición X, Y, Z del brazo derecho
- Posición X, Y, Z del brazo izquierdo

4.3.1.2. Esqueleto reconocido

Pensando en el estudio de la interacción con Leap Motion al esqueleto que se forma durante el desarrollo del reconocimiento incluye más información ofrecida por el dispositivo, de esta forma su análisis posterior será más sencillo.

- Posición X, Y, Z de la mano derecha.
- Posición X, Y, Z de la mano izquierda.
- Posición X, Y, Z del brazo derecho.
- Posición X, Y, Z del brazo izquierdo.
- Lista con las posiciones X, Y, Z de los dedos de la mano derecha.
- Lista con las posiciones X, Y, Z de los dedos de la mano izquierda.
- La mano empleada en el gesto: derecha, izquierda o ambas.

4.3.2. Grabador de gestos

Para actualizar el grabador de gestos se ha tenido que cambiar la información que se grababa. Los parámetros son el nombre del gesto, su identificador, el dispositivo empleado que será Leap Motion, la velocidad mínima y máxima del gesto, si participa la velocidad y qué ejes hay que tener en cuenta en el reconocimiento del gesto se almacenan inicialmente antes de iniciar la grabación. Durante la grabación de los gestos se almacenarán en el esqueleto las coordenadas X, Y, Z de las manos y las muñecas que se empleen en la definición del gesto.

4.3.3. Reconocimiento de gestos

Actualizar el reconocedor de los gestos implicaba añadir la nueva información obtenida del dispositivo Leap Motion. Para ello había que añadir la información que obtiene el escuchador en el buffer de datos que introduce las posiciones X, Y, Z de las manos, muñecas y dedos que detectaba Leap Motion.

Además se arreglaron varios bugs que tenía el reconocedor para realizar el cálculo de velocidades. Inicialmente se pensó que era un problema de concurrencia ya que hay dos hilos de ejecución distintos trabajando con los buffers (escuchador y comparador). Tras un estudio del código y realización de pruebas se comprobó que el error se encontraba en que el algoritmo NDtw era capaz de dar positivo con muy pocos *frames*, al tener el positivo se intentaba calcular la velocidad del gesto, pero este cálculo requiere al menos 20 *frames* y por eso daba errores. Para solucionarlo se implementó en el escuchador una condición para que solo los buffers con más de 25 *frames* sean enviados al comparador.

Además hubo que sacar el reconocedor de gestos del prototipo para poder introducirlo en el juego Interacttico, lo cual requería analizar múltiples funcionalidades para poder introducirlas correctamente al activar el dispositivo en el juego.

4.4. Grabación de los Gestos

Una vez actualizado el grabador de gestos con el nuevo SDK y añadiendo la información extra al esqueleto grabado, se procederá a grabar los gestos que han sido definidos en el apartado Definición de gestos. Para ello se empleará el grabador del prototipo actualizado (Figura 29). En él se introducen el nombre del gesto, identificador, si participa la velocidad y qué ejes participan (X, Y, Z). Con el primer gesto se creó un nuevo documento XML llamado Interacttico.xml donde se fueron añadiendo los demás gestos. Para el gesto rotar, debido a su complejidad del movimiento se configura el grabador (Figura 30) para aumenta el número de *frames* grabados para poder abarcar todo el gesto. Además permite abrir una aplicación web que muestra con un visualizador 3D el estado de la mano (Figura 31).

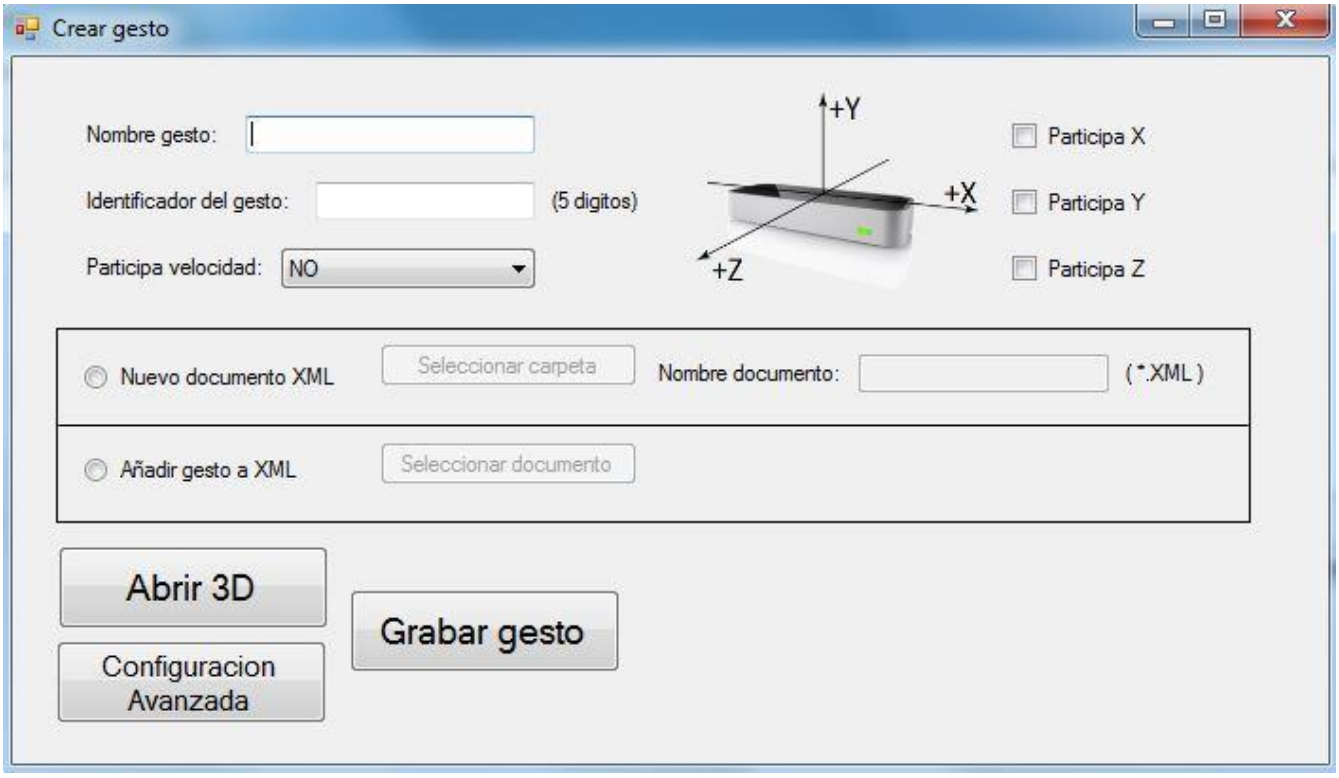

Figura 29 Grabación de Gestos

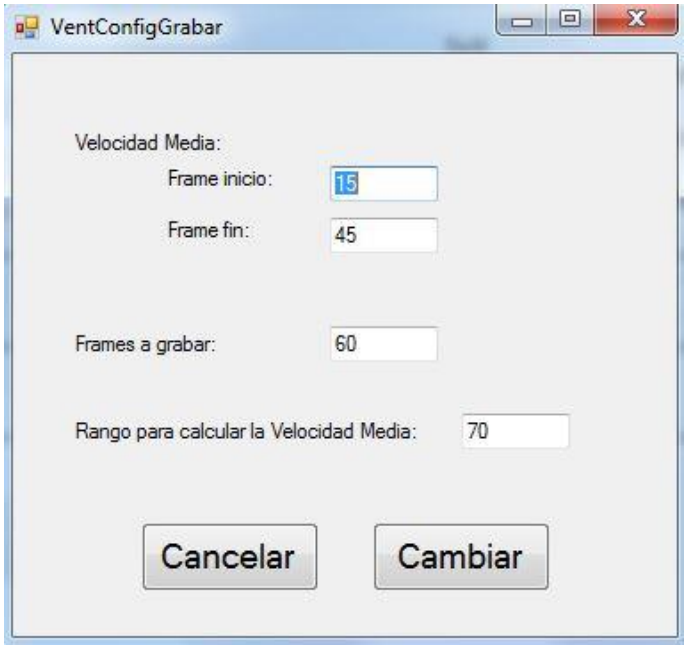

Figura 30 Configuración avanzada de grabar

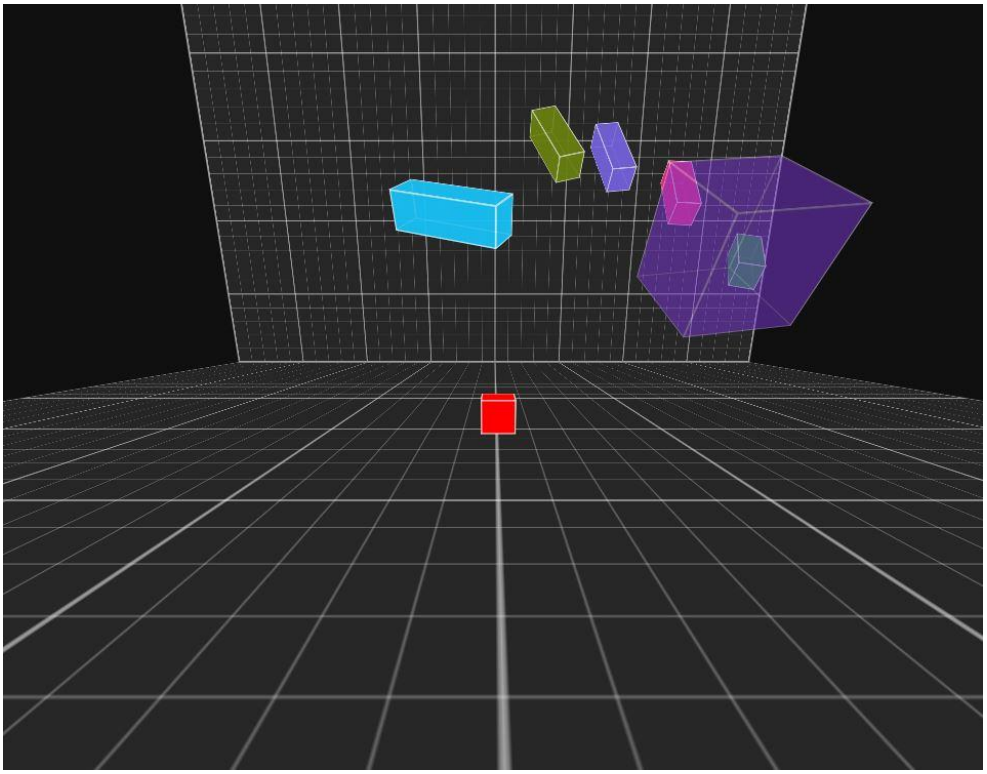

Figura 31 Visión 3D

4.5. Inserción del reconocedor en el juego Interacttico

Para la inserción del reconocedor en el juego Interacttico ha habido que realizar múltiples acciones en múltiples lenguajes. Para este caso influyen las siguientes partes:

- C#: en este lenguaje están programados el reconocedor de gestos de Leap Motion y el Wrapper que es el intermediario con la parte en ActionScript.
- ActionScript: aquí influyen el Wrapper que hace de intermediario con C# y la lógica del juego que controla las acciones que son correctas.
- Flash: aquí se encuentran las animaciones que se emplean en el juego y la implementación que las activa.

4.5.1. C#

En primer lugar hubo que separar el código del reconocedor del grabador de gestos empleado en el prototipo básico. Después sacar el código del reconocedor de gestos con el algoritmo NDtw se insertó en el código del proyecto C#. Dentro del reconocedor se modificó para que cuando reconociera uno de los gestos definidos enviara un mensaje al Wrapper de ActionScript con la acción que debe realizarse para que sea analizada por la lógica del juego.

También se incluyó en el Wrapper de C# el código necesario para comprobar que Leap Motion está disponible en el ordenador que se está ejecutando el juego cuando se ejecuta (hay que comprobar que hay un reconocedor en el ordenador y que hay un dispositivo conectado). Además de incluir el código para activar el reconocimiento de gestos cuando se selecciona el dispositivo Leap Motion en la pantalla de selección de dispositivo (Figura 32) y el de desactivar el reconocimiento cuando finaliza la partida. Dentro de Wrapper de C#, que tiene el reconocedor del teclado, también se permitió que pulsando las teclas “Control” y “O” abrir el menú de las opciones.

Además, para evitar problemas de reconocimiento debido a que el gesto Izquierda y el gesto Rotar inicialmente son similares, para evitar el problema se inhibe el reconocimiento del gesto Izquierda en la fase donde corresponde realizar el gesto Rotar para continuar correctamente el desarrollo del juego.

Figura 32 Selección de Dispositivo

4.5.2. Flash

La parte flash es la que tiene las imágenes y se encarga de realizar las animaciones correspondientes del juego. Ya que el funcionamiento del juego es idéntico para todos los dispositivos de entrada, la única diferencia que hay con lo que ya se encontraba disponible es que es necesario crear un botón (Figura 33) para seleccionar el dispositivo Leap Motion para emplearlo durante el juego.

Figura 33 Botón Leap Motion

4.5.3. ActionScript

En ActionScript se encuentran el Wrapper que recibe los gestos de los distintos reconocedores, entre ellos el de Leap Motion, y la lógica del juego. Para la recepción de los gestos se empleó el método ya implementado que recibe las acciones de Kinect y teclado. Se añadió un método para definir cuándo se encuentra Leap Motion activado.

Para la lógica del juego se añadieron las comprobaciones necesarias para saber qué acciones se pueden recibir en cada momento:

i. Fase de rotación: Rotar.
ii. Fase de detención de rotación: Seleccionar.
iii. Fase de apertura: Abrir.
iv. Fase de selección en la nube: Izquierda, Derecha y Seleccionar.
v. Fase de bajada: Bajar
vi. Fase de lanzamiento: Lanzar.

4.6. Generación del ejecutable

Para la ejecución del proyecto completo hay que agrupar los distintos elementos que interactúan con el juego de los diferentes lenguajes de programación. Además, para poder recoger la información de los dispositivos se necesitaba una base de datos en una carpeta que sea accesible para todos los usuarios del ordenador, para ello se empleó el directorio *%ProgramData%*, que es común para los usuarios y además ofrece permisos de lectura y escritura para poder trabajar con la base de datos (Figura 34). También se ha aprovechado esta carpeta para almacenar los gestos grabados que se emplean con Leap Motion en el archivo Interacttico.xml.

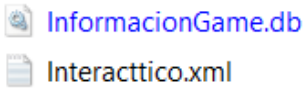

Figura 34 Contenido necesario en %ProgramData%

Los archivos y carpetas necesarios para la ejecución del programa son los siguientes (Figura 35):

- Audio: directorio que contiene los distintos audios que se reproducen durante la ejecución del juego.
- Lib: directorio con el reconocedor de la pantalla táctil y código de la parte Flash.
- Movimientos: los movimientos que deben realizar las animaciones del juego.
- Archivos realizados con C#:
    - a.xml: archivo que contiene los gestos grabados con Kinect.
    - AxInterop.ShockwaveFlashObjects.dll e Interop.ShockwaveFlashObjects.dll: librerías que permiten comunicarse con ActionScript.
    - Desinstalar.exe: Desinstala el juego borrando estos archivos.
    - JuegoCapaWPF.exe (ejecutable principal del juego), JuegoCapaWPF.exen, JuegoCapaWPF.vshost.exe y JuegoCapaWPF.vhost.exe.manifest: archivos generados con Visual Studio 2010
    - Leap.dll, LeapCSharp.dll y LeapCSharp.NET4.0.dll: librerías que emplea el reconocedor para Leap Motion obtenidas de la web de Leap Motion.
    - Microsoft Kinect.dll y Microsoft Kinect.xml
    - SQLite.dll, SQLite.NET.dll y SQLite3.dll: librerías para la base de datos.

- Archivos realizados con ActionScript:
  - CargadorFlash.html, CargadorFlash.swf: archivos generados con Adobe Flash Builder 4.6.
  - Esquema.xsd: archivo con los gestos que emplea la pantalla táctil.
- Flash:
  - Cargando.swf, pantallaInicial.swd, pantallaInicial.swf, swfobject.js, playerProductInstall.swf, y textLayout_2.0.0.232.swf contienen las acciones que permiten las animaciones que realiza el juego.

En el anexo adjunto al final se muestra la estructura completa del proyecto y cómo se generan estos archivos.

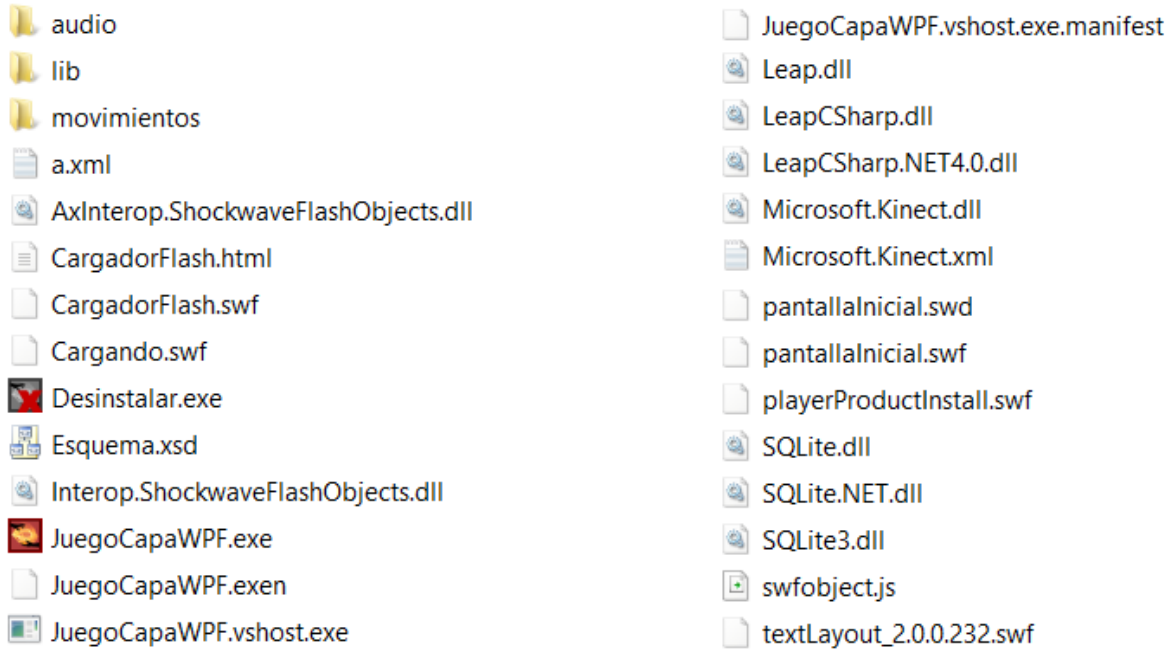

Figura 35 Archivos necesarios para la Ejecución

4.7. Pruebas

La realización de pruebas es una parte de gran importancia en todo proyecto. Para este se han tenido que realizar pruebas unitarias de los elementos modificados para ver que no tienen defectos antes de ser introducidos en el juego y pruebas de sistema para comprobar el correcto funcionamiento de todo el juego en conjunto:

4.7.1. Pruebas unitarias

4.7.1.1. Grabación de gestos

Para comprobar el correcto funcionamiento de la grabación de gestos se realizaron las siguientes pruebas:

- Abrir el grabador sin un dispositivo de Leap Motion conectado: al intentar abrir el grabador de gestos (Figura 29) sin el dispositivo conectado devolverá un mensaje de error y se mantendrá en la vista inicial (Figura 28).
- Grabar un gesto sin que se reconozca nada: si se intenta grabar un gesto sin que se reconozca nada el grabador se mantiene en espera hasta que reconozca al menos una mano para poder introducir información en el esqueleto grabado.
- Grabar un gesto sin movimiento: el grabador permite que la mano se mantenga en la misma posición durante la grabación, permitiendo reconocer un gesto donde la mano no se mueva en ninguna dirección durante un tiempo.
- Grabar un gesto correcto: la grabación de un gesto correcto generará el esqueleto correspondiente al gesto y lo almacenará en el archivo XML seleccionado antes de la grabación (ya sea nuevo o existiera previamente).

4.7.1.2. Reconocimiento de gestos

Una vez grabados los distintos gestos se realizaron pruebas para comprobar que todos los gestos se detectan correctamente.

- Gestos conocidos: el reconocedor devuelve el nombre del gesto correcto que se ha reconocido. Para esta prueba se reconocieron todos los gestos grabados (Definición de gestos) para el juego en Interacttico.xml: Izquierda (izq), Rotar (rot), Derecha (lan), Seleccionar (sel) y Bajar (baj).

- Gestos desconocidos: cuando el reconocedor no logra reconocer el gesto que se está realizando con la mano no se devuelve nada.

4.7.2. Pruebas de sistema

Para poder comprobar los cambios realizados en la lógica del juego y las modificaciones en flash es necesario incorporarlas al resto del sistema, ya que la lógica del juego necesita recibir las acciones desde el reconocedor y llamar a los elementos correspondientes en flash, motivo por el que no han tenido pruebas unitarias.

En el sistema completo se realizaron las siguientes pruebas:

- Ordenador sin Leap Motion instalado: al acceder al juego desde un ordenador que no tiene instalado los *drivers* de Leap Motion las funcionalidades son correctas, lo único que hace es dejar el botón del dispositivo Leap Motion en oscuro y desactivado ya que no se puede emplear ese dispositivo.
- Ordenador sin Leap Motion conectado: al iniciar el juego se comprueba que el ordenador tenga conectado algún dispositivo de Leap Motion, en este caso, al no tener ninguno conectado, al igual que en la prueba anterior se obtiene que el botón se encuentra oscuro y desactivado.
- Realización del juego empleando Leap Motion: en un ordenador con Leap Motion correctamente instalado y conectado el botón de selección de Leap Motion está disponible en todos los mini juegos. Al seleccionarlo el reconocedor de gestos se inicia correctamente y permite un funcionamiento completo durante la realización del juego. Cuando se reconoce un gesto en una fase que no corresponde el juego muestra el mensaje de error correspondiente y pide la repetición del gesto. En el caso de gesto correcto muestra la animación y continúa la siguiente fase correctamente.
- Realización del juego empleando otro dispositivo: también se han realizado pruebas de que funciona correctamente empleando otros dispositivos como la pantalla táctil (que tiene un sistema de selección especial) y el teclado. Las pruebas con ambos han mostrado un correcto funcionamiento de la lógica del juego también.

5. CONCLUSIONES

La realización de este Trabajo de Fin de Grado me ha permitido obtener nuevos conocimientos y reforzar otros adquiridos previamente. Entre estos nuevos conocimientos se encuentra el trabajo con el dispositivo de Leap Motion, desconocido para mí antes de la realización de este proyecto. Con el estudio de la API de este dispositivo y trabajo continuo el aprendizaje para programar con un nuevo dispositivo es más sencillo de lo que esperaba, requiere un gran trabajo pero con este dispositivo en particular la API ofrece las coordenadas reconocidas haciendo sencillo trabajar con números.

También me ha permitido reforzar los conocimientos de programación que tenía ya que para el desarrollo del reconocedor se trabaja con Tipos Abstractos de Datos (TADs) como los esqueletos, hay que trabajar con la concurrencia (el reconocedor tiene dos hilos: escuchador y comparador), la gestión de eventos (una vez reconocido un gesto enviar un evento a la lógica de juego), trabajar con múltiples lenguajes de programación a la vez (teniendo *wrappers* para comunicarse) y el estudio del código ya implementado.

Durante el desarrollo de este Trabajo de Fin de Grado he obtenido grandes mejoras sobre la redacción de documentación. La importancia de la realización de comentarios en el código para facilitar su entendimiento a próximos programadores, la correcta documentación de lo desarrollado, la realización de informes semanales, la creación de un plan de trabajo y la redacción de las distintas memorias necesarias para la entrega del Trabajo de Fin de Grado.

Al ser Interacttico un proyecto en el que trabajan varias personas, es necesario usar un control de versiones mediante un software de repositorio, que es un elemento importante del que no se habla nada en la carrera y que permite tener un orden y control del proyecto que con varias personas trabajando en el mismo proyecto lo vuelve imprescindible. El que he empleado en este trabajo es el cliente Tortoise para Windows que emplea SVN, que es el gestor de versiones.

Gracias a la realización del Practicum en el grupo de investigación CETTICO desarrollando la base de datos para almacenar la información recogida de los distintos dispositivos empleados en el proyecto Interacttico me permitió estudiar los lenguajes C# y ActionScript además de conocer detalles del proyecto como la lógica del juego y la comunicación entre lenguajes antes de la realización de este Trabajo de Fin de Grado.

Al ser un trabajo en grupo realizado en un laboratorio también he de hablar sobre las condiciones de trabajo. Las condiciones de trabajo con las que me he encontrado en este centro son muy buenas ya que el ambiente del grupo de trabajo es bueno, el trato con los compañeros es agradable y ameno. Mi tutor me ha apoyado con todas las dudas que me han ido surgiendo a lo largo de Trabajo de Fin de Grado y su guía para la realización de la memoria ha sido de gran ayuda. Finalmente se me ha proporcionado una gran flexibilidad de horarios para poder compaginar el trabajo con las clases, prácticas y exámenes de la facultad.

6. FUTURAS LÍNEAS DE TRABAJO

Este proyecto ofrece distintas líneas futuras de trabajo. En este caso distinguiremos entre tres tipos de líneas: un sistema que recoja la información de Leap Motion para su estudio, actualizaciones de Leap Motion, introducción de nuevos dispositivos de interacción en el juego y la inserción de nuevos mini juegos en el proyecto.

6.1. Recogida de información sobre Leap Motion

En trabajos previos se ha tratado sobre la realización de un Practicum en el grupo de investigación CETTICO en el que se guardaba la información que se consideró de interés sobre el teclado, la pantalla táctil y Kinect. Con Leap Motion se podría hacer otro Practicum similar en el que el alumno pueda aprender a trabajar con Leap Motion, diseñar a partir de la información definida en el apartado Definición de la información necesaria para el estudio de la interacción con Leap Motion las características que se quieren estudiar sobre Leap Motion y guardarlas en la base de datos.

6.2. Actualización de Leap Motion

Las últimas actualizaciones del SDK de Leap Motion se basan en la utilización de este dispositivo junto al de realidad virtual Oculus Rift (Figura 36) [12]. Se trabaja en mejoras de reconocimiento de Leap Motion montado en las gafas de Oculus Rift (Figura 37). La inclusión de la realidad virtual en el juego puede resultar muy llamativo para captar la atención de los niños con necesidades educativas especiales que son los usuarios a los que se quiere llegar con este proyecto.

Además todas las actualizaciones del SDK de Leap Motion siempre incluyen mejoras en el reconocimiento de la mano y la disminución de errores con el dispositivo. Estas mejoras siempre ayudan a mejorar el rendimiento de las aplicaciones que utilizan Leap Motion.

Figura 36 Dispositivo de realidad virtual Oculus Rift [13]

Figura 37 Oculus Rift con Leap Motion

6.3. Nuevos dispositivos de interacción

Se pueden introducir también nuevos dispositivos de interacción, actualmente se emplean teclado, pantalla táctil, Kinect y Leap Motion. Algunas opciones nuevas podrían ser:

- Ratón (Figura 38): es uno de los dispositivos de interacción típicos de los ordenadores y fáciles de entender.

Figura 38 Ratón

- Reconocimiento de voz: puede reconocer las órdenes mediante palabras, al moverse por la selección el juego ya dice la característica del elemento (la letra en deletrea, el nombre del objeto en agrupar conceptos y el color en bolos). De esta manera los usuarios con problemas visuales severos pueden interactuar con el ordenador de una manera más cómoda y sencilla.

6.4. Nuevos mini juegos

Se pueden introducir nuevos mini juegos al proyecto Interacttico. El objetivo de estos mini juegos es apoyar a la educación de niños con necesidades especiales, por lo que los juegos que se desarrollen deben ser educativos. Unos posibles conceptos para nuevos juegos podrían ser la identificación de formas (cuadrado, triángulo, círculo, etcétera), el aprendizaje de los números (aprender sobre su orden) o las operaciones matemáticas (sumas, restas, multiplicaciones, divisiones).

7. BIBLIOGAFRÍA

[1] Motion Control, Leap Motion [en línea]. Disponible en: https://www.leapmotion.com/ [Último acceso: 15/04/2015]

[2] Microsoft Windows, Kinect [en línea]. Disponible en: https://www.microsoft.com/en-us/kinectforwindows/ [Último acceso: 15/04/2015]

[3] La memoria final realizada para mi practicum que trata sobre el proyecto Interacttico (juego educativo para niños con necesidades educativas especiales).

[4] Redicals [en línea]. Disponible en: http://www.redicals.com/tech-gadgets-in-2014.html [Último acceso: 15/04/2015]

[5] La memoria final del trabajo de fin de carrera de Karim Laazizi Ruiz sobre Leap Motion.

[6] Algoritmo DTW [en línea]. Disponible en: http://www.psb.ugent.be/cbd/papers/gentxwarper/DTWalgorithm.htm [Último acceso: 15/04/2015]

[7] Implementación NDtw de Darjan Oblak [en línea]. Disponible en: https://github.com/doblak/ndtw [Último acceso 15/04/2015]

[8] Novedades SDK 2.2.2.24469 [en línea]. Disponible en https://community.leapmotion.com/t/v2-software-consumer-sdk-consumer-update-v2-2-2-24469/2391 [Último acceso: 15/04/2015]

[9] Foro para desarrolladores de Leap Motion [en línea]. Disponible en: https://community.leapmotion.com/c/development [Último acceso: 15/04/2015]

[10] Novedades SDK 2.1.2.21921 [en línea]. Disponible en: https://community.leapmotion.com/t/v2-skeletal-tracking-beta-new-sdk-and-build-v2-1-2-21921/1705 [Último acceso: 15/04/2015]

[11] Documentación sobre imágenes de Leap Motion [en línea]. Disponible en: https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Images.html [Último acceso: 15/04/2015]

[12] Página web oficial del dispositivo de realidad virtual Oculus Rift [en línea]. Disponible en: https://www.oculus.com/es/ [Último acceso: 20/06/2015]

[13] Página web para compras del dispositivo de realidad virtual Oculus Rift [en línea]. Disponible en: https://www1.oculus.com/order/ [Último acceso: 20/06/2015]

[14] Página web para compras de dispositivos informáticos [en línea]. Disponible en: http://www.pccomponentes.com/natec_diver_black_optical_mouse.html [Último acceso: 20/06/2015]

8. Anexo: Estructura de directorios

8.1. Estructura del prototipo básico de Leap Motion

El prototipo básico de Leap Motion se divide en los siguientes directorios (Figura 39):

- Algoritmo: esta carpeta contiene la implementación del algoritmo NDtw [7].
- Consola: en esta carpeta se guardan pruebas básicas sobre el correcto funcionamiento de las diferentes funcionalidades.
- Estructura: contiene las clases que leen y generan los distintos esqueletos que empleamos para reconocer y grabar gestos.
- GestorGesto: contiene las clases que leen y generan los archivos donde se generan los gestos.
- Imágenes: contiene las distintas imágenes que se emplean en las vistas de la aplicación.
- Recognizer: contiene las clases que generan el buffer circular, el hilo escuchador y el hilo comparador que permiten reconocer los gestos.
- Recorder: contiene la clase que graba los gestos realizados con el dispositivo Leap Motion.
- Ventana: Contiene las distintas vistas que se emplean durante el uso de la aplicación.

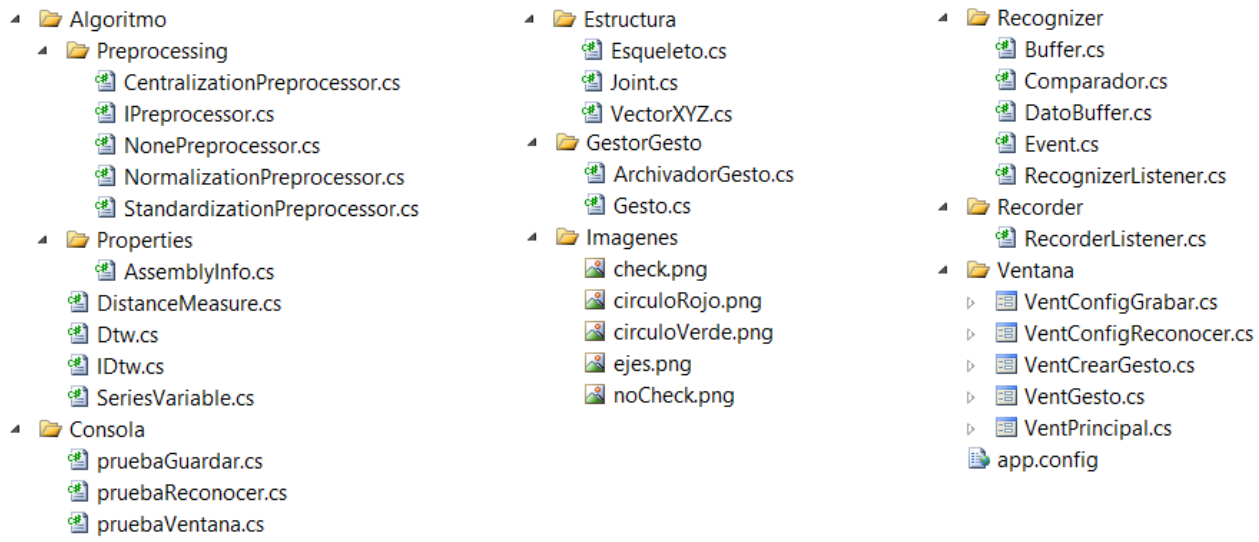

Figura 39 Estructura Leap Motion

8.2. Estructura Interacttico

El proyecto Interacttico tiene una estructura más compleja (Figura 40):

- Audios: contiene los audios que se reproducen durante el juego.
- Base de Datos del juego: contiene una copia de la base de datos con las pruebas realizadas con el juego.
- Bin: este directorio contiene otro con el nombre Debug donde se introducen los archivos generados por el proyecto JuegoCapaWPF.sln realizado en C# con Visual Studio 2010 Professional. Aquí se encuentran JuegoCapaWPF.exe (ejecutable principal del juego), JuegoCapaWPF.exen, JuegoCapaWPF.vshost.exe y JuegoCapaWPF.vhost.exe.manifest.
- CargadorFlash: contiene el código en ActionScript de la lógica del juego y el reconocedor de la pantalla táctil que se editan empleando Adobe Flash Builder 4.6 y generan su salida en la subcarpeta bin-debug.
- Com: contiene las clases con las acciones Flash que se emplean para las animaciones del juego realizadas con Adobe Flash Profesional CS3.
- Dlls para WPF: contiene las librerías que emplea C# para comunicarse con los reconocedores, la base de datos y ActionScript.
- Imagenes: contiene las distintas imágenes que emplea el juego para realizar las animaciones.
- Instalador y Desinstalador: contiene dos proyectos en C#:
    - Instalador: proyecto realizado en Visual Studio 2010 Professional para instalar el juego en un ordenador.
    - Desinstalador: elimina el juego instalado de Archivos de Programa. Genera Desinstalador.exe.
- JuegoCapaWPF: contiene el proyecto C# con la implementación sobre los reconocedores y la conexión con la base de datos. Este proyecto se abre con Visual Studio 2010 Professional desde JuegoCapaWPF.sln. La estructura de este subproyecto se muestra en la Figura 41. Hay similitudes con la Figura 39 ya que hay partes que se han tomado del prototipo básico de Leap Motion, el cuál es un proyecto independiente con sus funcionalidades definidas, y realizado modificaciones para que funcione el reconocedor en el juego.
- Lib: es la librería con el código generado por Adobe Flash Professional CS3.

- Movimientos: contiene los distintos movimientos que se emplean en las animaciones del juego.
- Temp: contiene los archivos temporales que genera Visual Studio 2010 Professional.
- Cargando.fla, cargando.psd y pantallaInicial.fla contienen las vistas que se emplean en el juego. Se modifica empleando Adobe Flash Professional CS3.
- pantallaInicial.fla.bak es una copia de seguridad que genera AdobeFlash Professional CS3.

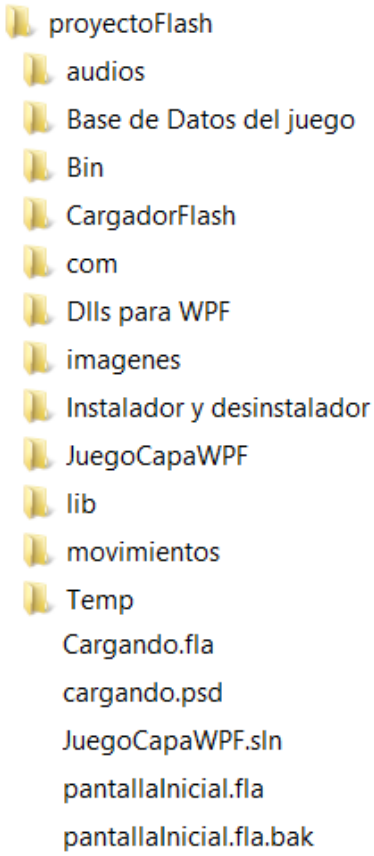

Figura 40 Estructura Interacttico

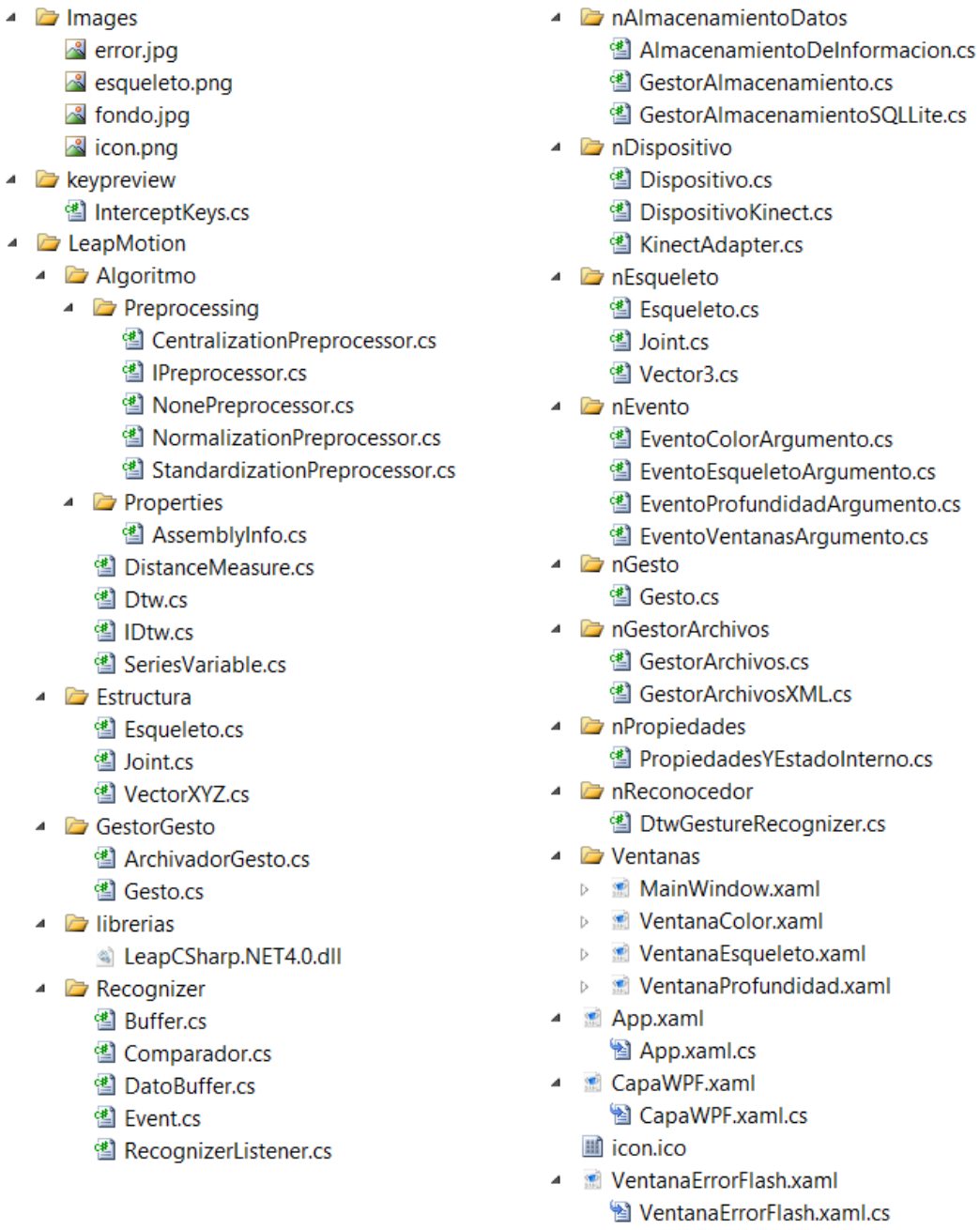

Figura 41 Esquema JuegoCapaWPF

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=Facultad de Informatica - UPM, C=ES
	Fecha/Hora	Thu Jun 25 13:02:39 CEST 2015
	Emisor del Certificado	EMAILADDRESS=camanager@fi.upm.es, CN=CA Facultad de Informatica, O=Facultad de Informatica - UPM, C=ES
	Numero de Serie	630
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)